```
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT        SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT        SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT        SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT        SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT        SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT        SSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO        TTT           SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO        TTT           SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO        TTT           SSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT                    SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT                    SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT                    SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT                    SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT                    SSS
BBB         BBB  OOO         OOO  OOO         OOO        TTT                    SSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO           TTT        SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO           TTT        SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO           TTT        SSSSSSSSSSSS
```

F 16

```
**FILE**ID**SHARE
```

```
  SSSSSSSS  HH      HH    AAAAAA    RRRRRRRR    EEEEEEEEEE
  SSSSSSSS  HH      HH    AAAAAA    RRRRRRRR    EEEEEEEEEE
SS          HH      HH  AA      AA  RR      RR  EE
SS          HH      HH  AA      AA  RR      RR  EE
SS          HH      HH  AA      AA  RR      RR  EE
SS          HH      HH  AA      AA  RR      RR  EE
  SSSSSS    HHHHHHHHHH  AA      AA  RRRRRRRR    EEEEEEEE
  SSSSSS    HHHHHHHHHH  AA      AA  RRRRRRRR    EEEEEEEE
        SS  HH      HH  AAAAAAAAAA  RR  RR      EE
        SS  HH      HH  AAAAAAAAAA  RR  RR      EE
        SS  HH      HH  AA      AA  RR    RR    EE              ....
        SS  HH      HH  AA      AA  RR    RR    EE              ....
SSSSSSSS    HH      HH  AA      AA  RR      RR  EEEEEEEEEE      ....
SSSSSSSS    HH      HH  AA      AA  RR      RR  EEEEEEEEEE      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000          .TITLE  SHARE   SHARED MEMORY INITIALIZATION
0000          .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ; FACILITY:     SYSGEN
0000    ;
0000    ; ABSTRACT:     THIS MODULE INITIALIZES AND CONNECTS THE PROCESSOR TO
0000    ;               A PORT OF A MULTI-PORT (SHARED) MEMORY.
0000    ;
0000    ;
0000    ; ENVIRONMENT:  NATIVE/USER MODE, PRIVILEGED
0000    ;
0000    ; AUTHOR:       LEN KAWELL, CREATION DATE: 19-DEC-1978
0000    ;
0000    ; MODIFICATION HISTORY:
0000    ;
0000    ;       V03-007 KPL00100        Peter Lieberwirth       10-Feb-1984
0000    ;               Use longword format CONFREGL due to impending BI devices
0000    ;               having 16-bit device types.
0000    ;
0000    ;       V03-006 WHM00001        Bill Matthews           14-Dec-1983
0000    ;               Change references to ACF$B_CUNIT to ACF$W_CUNIT
0000    ;
0000    ;       V03-005 KDM42758        Kathleen D. Morse       04-Jan-1983
0000    ;               Minimize the shared memory structure quotas with their
0000    ;               maximums (e.g., SHD$W_MBXQUOTA with SHD$W_MBXMAX).
0000    ;               During a CONNECT, subtract one from the SHD$W_MBXQUOTA and
0000    ;               the SHD$W_CEFQUOTA counts for each structure owned by this port.
0000    ;
0000    ;
0000    ;--
```

```
          0000           .SBTTL  DECLARATIONS
          0000   ;
          0000   ; INCLUDE FILES:
          0000   ;
          0000
          0000   ;
          0000   ; MACROS:
          0000   ;
          0000
          0000   ;
          0000   ; PUT_OUTPUT - MACRO TO FORMAT AND PUT A MESSAGE TO SYS$OUTPUT
          0000   ;
          0000           .MACRO  PUT_OUTPUT MSG,ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
          0000           .SAVE   LSB
          0000           .PSECT  NONPAGED_DATA   RD,WRT,NOEXE,QUAD
          0000   $$DESC=.
          0000           .ASCID  \MSG\
          0000           .RESTORE
          0000
          0000           .IF NB ARG1                             ; IF FORMATTING NEEDED
          0000           MOVAB   -128(SP),SP                     ; ALLOCATE FORMAT BUFFER
          0000           PUSHL   SP                              ; CREATE BUFFER DESCRIPTOR
          0000           PUSHL   #128                            ; ...
          0000           MOVL    SP,R0                           ; GET ADDR OF DESCRIPTOR
          0000           $FAO_S  $$DESC,(R0),(R0),-              ; FORMAT THE OUTPUT
          0000                   ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
          0000           PUSHL   SP                              ; SET ADDR OF BUFFER DESC
          0000           CALLS   #1,G^LIB$PUT_OUTPUT             ; OUTPUT THE FORMATTED TEXT
          0000           MOVAB   128+8(SP),SP                    ; DEALLOCATE BUFFER AND DESC
          0000
          0000           .IFF
          0000           PUSHAQ  $$DESC                          ; SET ADDR OF TEXT DESC
          0000           CALLS   #1,G^LIB$PUT_OUTPUT             ; OUTPUT THE TEXT
          0000           .ENDC
          0000
          0000           .ENDM   PUT_OUTPUT
          0000
          0000   ;
          0000   ; EQUATED SYMBOLS:
          0000   ;
          0000
08F0D180  0000   INITLOCK_TIMOUT = 15*10*1000*1000       ; INITIALIZATION LOCK TIMEOUT TIME
02FAF080  0000   INITPOLL_TIMOUT = 5*10*1000*1000        ; INITIALIZATION POLL TIMEOUT TIME
          0000
          0000   ;
          0000   ; SYSTEM DEFINITIONS
          0000   ;
          0000           $ACBDEF                         ; AST CONTROL BLOCKS
          0000           $ACFDEF                         ; CONFIGURATION CONTROL BLOCK
          0000           $ADPDEF                         ; NEXUS ADAPTER CONTROL BLOCKS
          0000           $CEBDEF                         ; COMMON EVENT FLAG BLOCKS
          0000           $DYNDEF                         ; DYNAMIC DATA STRUCTURE TYPE CODES
          0000           $GSDDEF                         ; GLOBAL SECTION DESCRIPTOR
          0000           $IPLDEF                         ; INTERRUPT PRIORITY LEVELS
          0000           $MBXDEF                         ; MAILBOX CONTROL BLOCK
          0000           $MPMDEF                         ; MULTIPORT MEMORY ADAPTER
          0000           $NDTDEF                         ; NEXUS DEVICE TYPES
```

```
                 0000           $PRDEF                          ; PROCESSOR REGISTERS
                 0000           $PRQDEF                         ; INTER-PROCESSOR REQUESTS
                 0000           $PTEDEF                         ; PAGE TABLE ENTRIES
                 0000           $RPBDEF                         ; RESTART PARAMETER BLOCK
                 0000           $RSNDEF                         ; RESOURCE NUMBERS
                 0000           $SHBDEF                         ; SHARED MEMORY CONTROL BLOCK
                 0000           $SHDDEF                         ; SHARED MEMORY DATAPAGE
                 0000           $SSDEF                          ; SYSTEM ERROR CODES
                 0000           $STSDEF                         ; STATUS CODES
                 0000           $SYSGMSGDEF                     ; SYSGEN MESSAGES
                 0000           $TPADEF                         ; TPARSE
                 0000           $VADEF                          ; VIRTUAL ADDRESSES
                 0000
                 0000   ;
                 0000   ; OWN STORAGE:
                 0000   ;
             00000000           .PSECT  NONPAGED_DATA   RD,WRT,NOEXE,QUAD
                 0000
                 0000   SHR_VALUES:                             ; START OF QUALIFIER VALUES
                 0000   SHR_Q_MEMNAME:                          ; MEMORY NAME DESCRIPTOR
        00000008 0000           .BLKL   2
                 0008   SHR_W_UNIT:                             : MEMORY UNIT #
        0000000A 0008           .BLKW   1
                 000A   SHR_W_GBLCNT:                           ; GLOBAL SECTION COUNT
        0000000C 000A           .BLKW   1
                 000C   SHR_W_MBXCNT:                           ; MAILBOX COUNT
        0000000E 000C           .BLKW   1
                 000E   SHR_W_CEFCNT:                           ; COMMON EVENT FLAG CLUSTER COUNT
        00000010 000E           .BLKW   1
                 0010   SHR_W_GBLQUO:                           ; GLOBAL SECTION QUOTA FOR PORT
        00000012 0010           .BLKW   1
                 0012   SHR_W_MBXQUO:                           ; MAILBOX QUOTA  FOR PORT
        00000014 0012           .BLKW   1
                 0014   SHR_W_CEFQUO:                           ; COM EVT FLAG CLUSTER QUOTA FOR PORT
        00000016 0014           .BLKW   1
                 0016   SHR_L_POOLBCNT:                         ; POOL BLOCK COUNT
        0000001A 0016           .BLKL   1
                 001A   SHR_L_POOLBSIZ:                         ; POOL BLOCK SIZE
        0000001E 001A           .BLKL   1
                 001E   SHR_L_PRQCNT:                           ; INTER-PROCESSOR REQUEST COUNT
        00000022 001E           .BLKL   1
                 0022   SHR_L_START:                            ; STARTING PFN
        00000026 0022           .BLKL   1
                 0026   SHR_B_OPTIONS:                          ; COMMAND OPTIONS
        00000027 0026           .BLKB   1
                 0027           _VIELD  SHR_OPT,0,<-            ;  OPTION DEFINITIONS
                 0027                   <INIT,,M>,-             ;   INITIALIZE MEMORY
                 0027                   >
                 0027
                 0027
                 0027   SHR_L_MEMSIZE:                          ; SIZE OF SHARED MEMORY (PAGES)
        0000002B 0027           .BLKL   1
                 002B
                 002B   SHR_L_MEMPFN:                           ; STARTING PFN OF MEMORY
        0000002F 002B           .BLKL   1
                 002F
                 002F   SHR_L_GSDSIZE:                          ; SIZE OF A GLOBAL SECTION DESC
```

```
                           00000033  002F                 .BLKL   1
                                     0033
                                     0033  SHR_L_CEFSIZE:                               ; SIZE OF SHMCEB, MASTER COMM EVT BLOCK
                           00000037  0033                 .BLKL   1
                                     0037
                                     0037  SHR_L_DATAPAGE:                              ; ADDRESS OF DATAPAGE
                           0000003B  0037                 .BLKL   1
                                     003B
                                     003B  SHR_L_SHDPTE:                                ; ADDRESS OF DATAPAGE PTE
                           0000003F  003B                 .BLKL   1
                                     003F
                                     003F  SHR_L_ADP:                                   ; ADPATER CONTROL BLOCK ADDRESS
                           00000043  003F                 .BLKL   1
                                     0043
                                     0043  SHR_T_MBDEVNAME:                             ; MAILBOX DEVICE NAME
                        42 42 4D 00' 0043                 .ASCIC  /MBB/
                                  03 0043
                                     0047
                               00000000                   .PSECT  NONPAGED_CODE   RD,NOWRT,EXE,LONG ; PURE DATA SECTION
                                     0000
                                     0000  SHR_T_MBDRVNAME:                             ; MAILBOX DRIVER NAME
      52 45 56 49 52 44 58 42 4D 00' 0000                 .ASCIC  /MBXDRIVER/
                                  09 0000
                                     000A
                                     000A  ;
                                     000A  ; AUTODIN-II POLYNOMIAL TABLE
                                     000A  ;
                                     000A
                                     000A  AUTODIN:
26D930AC 3B6E20C8 1DB71064 00000000  000A                 .LONG   ^000000000000,^003555610144,^007333420310,^004666230254
5005713C 4DB26158 6B6B51F4 76DC4190  001A                 .LONG   ^016667040620,^015332650764,^011554460530,^012001270474
CB61B38C D6D6A3E8 F00F9344 EDB88320  002A                 .LONG   ^035556101440,^036003711504,^032665521750,^031330331614
BDBDF21C A00AE278 86D3D2D4 9B64C2B0  003A                 .LONG   ^023331141260,^020664751324,^024002561170,^027557371034
                                     004A
```

```
                              004A          .SBTTL  SHARE COMMAND QUALIFIER ACTION ROUTINES
                              004A    ;++
                              004A    ; FUNCTIONAL DESCRIPTION:
                              004A    ;
                              004A    ;       THESE TPARSE ACTION ROUTINES STORE THE SHARE COMMAND QUALIFIER
                              004A    ;       VALUES AND CHECK THEIR VALIDITY.
                              004A    ;
                              004A    ;
                              004A    ; CALLING SEQUENCE:
                              004A    ;
                              004A    ;       CALLED AS A TPARSE ACTION ROUTINE.
                              004A    ;       (SEE THE RUN-TIME LIBRARY MANUAL FOR DETAILS)
                              004A    ;
                              004A    ; INPUTS:
                              004A    ;
                              004A    ;       STANDARD TPARSE PARAMETER BLOCK.
                              004A    ;
                              004A    ; OUTPUTS:
                              004A    ;
                              004A    ;       VALUES STORED FOR FUTURE USE BY SHARE COMMAND PROCESSING.
                              004A    ;
                              004A    ;       IF A VALUE IS INVALID, FAILURE IS RETURNED TO PRODUCE A SYNTAX
                              004A    ;       ERROR AND STOP PROCESSING.
                              004A    ;
                              004A    ;
                              004A    ;--
                          0000004A          .PSECT  NONPAGED_CODE   RD,NOWRT,EXE,LONG
                              004A          .DEFAULT DISPLACEMENT WORD      ; DEFAULT PC DISPLACEMENT
                              004A
                              004A
                              004A    GEN$SHR_RESET::                          ; RESET QUALIFIER VALUES
                        0000  004A          .WORD   0                        ; ENTRY MASK
        000A'CF    20     B0  004C          MOVW    #32,SHR_W_GBLCNT         ; SET DEFAULT GSD COUNT
        000C'CF    20     B0  0051          MOVW    #32,SHR_W_MBXCNT         ; SET DEFAULT MAILBOX COUNT
        000E'CF    20     B0  0056          MOVW    #32,SHR_W_CEFCNT         ; SET DEFAULT COM EV FLG CLUSTER COUNT
   0010'CF    7FFF 8F     B0  005B          MOVW    #32767,SHR_W_GBLQUO      ; SET DEFAULT GSD QUOTA FOR PORT
   0012'CF    7FFF 8F     B0  0062          MOVW    #32767,SHR_W_MBXQUO      ; SET DEFAULT MBX QUOTA FOR PORT
   0014'CF    7FFF 8F     B0  0069          MOVW    #32767,SHR_W_CEFQUO      ; SET DEFAULT CEF QUOTA FOR PORT
   001A'CF    0080 8F     3C  0070          MOVZWL  #128,SHR_L_POOLBSIZ      ; SET DEFAULT POOL BLOCK SIZE
   0016'CF    0080 8F     3C  0077          MOVZWL  #128,SHR_L_POOLBCNT      ; SET DEFAULT POOL BLOCK COUNT
   001E'CF    0040 8F     3C  007E          MOVZWL  #64,SHR_L_PRQCNT         ; SET DEFAULT PRQ COUNT
              0022'CF     D4  0085          CLRL    SHR_L_START              ; SET DEFAULT STARTING PFN
              0026'CF     94  0089          CLRB    SHR_B_OPTIONS            ; RESET ALL OPTIONS
              50     01   D0  008D          MOVL    #1,R0                    ; SET SUCCESS
                          04  0090          RET                              ; RETURN
                              0091
                              0091    GEN$SHR_MEMNAME::                        ; SET THE MEMORY NAME
                        0000  0091          .WORD   0                        ; ENTRY MASK
         0F      10 AC    B1  0093          CMPW    TPA$L_TOKENCNT(AP),#15   ; IS NAME TOO LONG?
                 07       1E  0097          BGEQU   10$                      ; BRANCH IF YES
     0000'CF     10 AC    7D  0099          MOVQ    TPA$L_TOKENCNT(AP),SHR_Q_MEMNAME ; SET MEMORY NAME DESC
                          04  009F          RET                              ; EXIT
                              00A0    10$:
                 50       D4  00A0          CLRL    R0                       ; SET FAILURE
                          04  00A2          RET                              ; EXIT
                              00A3
                              00A3    GEN$SHR_UNIT::                           ; SET MEMORY UNIT #
```

```
                  0000  00A3              .WORD   0                              ; ENTRY MASK
0008'CF    1C AC    B0  00A5              MOVW    TPA$L_NUMBER(AP),SHR_W_UNIT   ; SET MEMORY UNIT #
                    04  00AB              RET                                    ; EXIT
                        00AC
                        00AC      GEN$SHR_GBLCNT::                               ; SET GLOBAL SECTION COUNT
                  0000  00AC              .WORD   0                              ; ENTRY MASK
000A'CF    1C AC    B0  00AE              MOVW    TPA$L_NUMBER(AP),SHR_W_GBLCNT ; SET GLOBAL SECTION CNT
              05    12  00B4              BNEQ    10$                            ; BRANCH IF AT LEAST 1
  000A'CF     01    B0  00B6              MOVW    #1,SHR_W_GBLCNT                ; SET MINIMUM OF 1
                        00BB      10$:
                    04  00BB              RET                                    ; EXIT
                        00BC
                        00BC      GEN$SHR_MBXCNT::                               ; SET MAILBOX COUNT
                  0000  00BC              .WORD   0                              ; ENTRY MASK
000C'CF    1C AC    B0  00BE              MOVW    TPA$L_NUMBER(AP),SHR_W_MBXCNT ; SET MAILBOX COUNT
              05    12  00C4              BNEQ    10$                            ; BRANCH IF AT LEAST 1
  000C'CF     01    B0  00C6              MOVW    #1,SHR_W_MBXCNT                ; SET MINIMUM OF 1
                        00CB      10$:
                    04  00CB              RET                                    ; EXIT
                        00CC
                        00CC      GEN$SHR_CEFCNT::                               ; SET COMMON EVENT FLAG CLUSTER COUNT
                  0000  00CC              .WORD   0                              ; ENTRY MASK
000E'CF    1C AC    B0  00CE              MOVW    TPA$L_NUMBER(AP),SHR_W_CEFCNT ; SET COM EVT FLAG CLUSTER COUNT
              05    12  00D4              BNEQ    10$                            ; BRANCH IF AT LEAST 1
  000E'CF     01    B0  00D6              MOVW    #1,SHR_W_CEFCNT                ; SET MINIMUM OF 1
                        00DB      10$:
                    04  00DB              RET                                    ; EXIT
                        00DC
                        00DC      GEN$SHR_GBLMAX::                               ; SET PORT MAX GLOBAL SECTIONS
                  0000  00DC              .WORD   0                              ; ENTRY MASK
0010'CF    1C AC    B0  00DE              MOVW    TPA$L_NUMBER(AP),SHR_W_GBLQUO ; SET PORT MAX
                    04  00E4              RET                                    ; EXIT
                        00E5
                        00E5      GEN$SHR_MBXMAX::                               ; SET PORT MAX MAILBOXES
                  0000  00E5              .WORD   0                              ; ENTRY MASK
0012'CF    1C AC    B0  00E7              MOVW    TPA$L_NUMBER(AP),SHR_W_MBXQUO ; SET PORT MAX
                    04  00ED              RET                                    ; EXIT
                        00EE
                        00EE      GEN$SHR_CEFMAX::                               ; SET PORT MAX COM EVT FLG CLUSTERS
                  0000  00EE              .WORD   0                              ; ENTRY MASK
0014'CF    1C AC    B0  00F0              MOVW    TPA$L_NUMBER(AP),SHR_W_CEFQUO ; SET PORT MAX
                    04  00F6              RET                                    ; EXIT
                        00F7
                        00F7      GEN$SHR_POOLC::                                ; SET POOL BLOCK COUNT
                  0000  00F7              .WORD   0                              ; ENTRY MASK
0016'CF    1C AC    D0  00F9              MOVL    TPA$L_NUMBER(AP),SHR_L_POOLBCNT ; SET POOL BLOCK COUNT
              05    12  00FF              BNEQ    10$                            ; BRANCH IF NOT = 0
  0016'CF     01    D0  0101              MOVL    #1,SHR_L_POOLBCNT              ; SET MINIMUM OF 1
                        0106      10$:
                    04  0106              RET                                    ; EXIT
                        0107
                        0107      GEN$SHR_POOLS::                                ; SET POOL BLOCK SIZE
                  0000  0107              .WORD   0                              ; ENTRY MASK
     50   001A'CF   DE  0109              MOVAL   SHR_L_POOLBSIZ,R0              ; GET ADDR OF SIZE BUFFER
       60   1C AC   D0  010E              MOVL    TPA$L_NUMBER(AP),(R0)          ; SET SPECIFIED POOL BLOCK SIZE
         51    1C   D0  0112              MOVL    #<ACB$L_KAST+4>,R1             ; GET MINIMUM SIZE (SIZE OF ACB)
         51    60   D1  0115              CMPL    (R0),R1                        ; IS SPECIFIED SIZE BIG ENOUGH?
```

```
                 03   1E  0118          BGEQU   10$                             ; BRANCH IF YES
            50   51   D0  011A          MOVL    R1,R0                           ; ELSE, SET SIZE TO MINIMUM
                          011D  10$:
            60   07   C0  011D          ADDL    #^B111,(R0)                     ; ROUND UP FOR QUADWORD ALIGNMENT
            60   07   CA  0120          BICL    #^B111,(R0)                     ; ...
            50   01   D0  0123          MOVL    #SS$_NORMAL,R0                  ; RETURN SUCCESS
                      04  0126          RET                                     ; EXIT
                          0127
                          0127  GEN$SHR_PRQCNT::                                ; SET INTER-PROCESSOR REQUEST BLOCK COUNT
                    0000  0127          .WORD   0                               ; ENTRY MASK
   001E'CF     1C AC  D0  0129          MOVL    TPA$L_NUMBER(AP),SHR_L_PRQCNT   ; SET PRQ COUNT
                 05   12  012F          BNEQ    10$                             ; BRANCH IF NOT = 0
      001E'CF    01   D0  0131          MOVL    #1,SHR_L_PRQCNT                 ; SET MINIMUM OF 1
                          0136  10$:
                      04  0136          RET
                          0137
                          0137  GEN$SHR_START::                                 ; SET START OF MEMORY
                    0000  0137          .WORD   0                               ; ENTRY MASK
   0022'CF     1C AC  D0  0139          MOVL    TPA$L_NUMBER(AP),SHR_L_START    ; SET START OF MEMORY PFN
                      04  013F          RET                                     ; EXIT
                          0140
                          0140  GEN$SHR_INIT::                                  ; SET INIT OPTION
                    0000  0140          .WORD   0                               ; ENTRY MASK
      0026'CF    01   88  0142          BISB    #SHR_OPT_M_INIT,SHR_B_OPTIONS   ; SET INIT OPTION
                      04  0147          RET                                     ; EXIT
                          0148
```

```
                              0148          .SBTTL  SHARE COMMAND MAIN ACTION ROUTINE
                              0148    ;++
                              0148    ; FUNCTIONAL DESCRIPTION:
                              0148    ;
                              0148    ;       THIS IS THE MAIN SHARE COMMAND ACTION ROUTINE.  IT PERFORMS
                              0148    ;       ALL THE REAL WORK OF INITIALIZING AND/OR CONNECTING TO A SHARED
                              0148    ;       MEMORY.
                              0148    ;
                              0148    ;
                              0148    ; CALLING SEQUENCE:
                              0148    ;
                              0148    ;       CALLED AS A TPARSE ACTION ROUTINE.
                              0148    ;       (SEE THE RUN-TIME LIBRARY MANUAL FOR DETAILS)
                              0148    ;
                              0148    ; INPUTS:
                              0148    ;
                              0148    ;       STANDARD TPARSE PARAMETER BLOCK.
                              0148    ;
                              0148    ;       QUALIFIER VALUES ASSUMED TO BE STORED BY PREVIOUS ACTION
                              0148    ;       ROUTINES.
                              0148    ;
                              0148    ; OUTPUTS:
                              0148    ;
                              0148    ;       PROCESSOR CONNECTED TO THE SHARED MEMORY.  IF /INIT IS SPECIFIED,
                              0148    ;       THE MEMORY AND DATASTRUCTURES ARE ALSO INITIALIZED.
                              0148    ;
                              0148    ;--
                              0148
                              0148   GEN$SHARE::                                  ; MAIN SHARE ACTION ROUTINE
                        0000  0148          .WORD   0                             ; ENTRY MASK
                              014A
                              014A          $CMKRNL_S SHARE                       ; DO IT IN KERNEL MODE
              19 50     E9    0157          BLBC    R0,10$                        ; BRANCH IF FAILURE
                              015A          $CMEXEC_S SHOW_STRUCT                 ; SHOW THE STRUCTURES
              09 50     E9    0167          BLBC    R0,10$                        ; BRANCH IF FAILURE
         083C'CF   00   FB    016A          CALLS   #0,LOADMBDRIVER               ; LOAD THE MAILBOX DRIVER
              01 50     E9    016F          BLBC    R0,10$                        ; BRANCH IF FAILURE
                        04    0172          RET                                   ; EXIT
                              0173
                              0173   10$:
                   02   F0    0173          INSV    #STS$K_ERROR,-                ; CONVERT STATUS TO ERROR
          50   03  00         0175                  #STS$V_SEVERITY,#STS$S_SEVERITY,R0
                   50   DD    0178          PUSHL   R0                            ; SET ERROR
     00000000'GF   01   FB    017A          CALLS   #1,G^LIB$SIGNAL               ; SIGNAL THE ERROR
              50   01   D0    0181          MOVL    #STS$K_SUCCESS,R0             ; SET SUCCESS FOR PARSER
                        04    0184          RET                                   ; EXIT
                              0185
```

```
                                  0185          .SBTTL  SHARE KERNEL ROUTINE
                                  0185  ;++
                                  0185  ;
                                  0185  ; SHARE - KERNEL ROUTINE TO INIT AND CONNECT TO A SHARED MEMORY
                                  0185  ;
                                  0185  ; CALLING SEQUENCE:
                                  0185  ;
                                  0185  ;       $CMKRNL_S SHARE
                                  0185  ;
                                  0185  ; INPUTS:
                                  0185  ;
                                  0185  ;       SHARE COMMAND QUALIFIER VALUES STORED.
                                  0185  ;
                                  0185  ; OUPUTS:
                                  0185  ;
                                  0185  ;       R0 = SUCCESS OR FAILURE STATUS.
                                  0185  ;
                                  0185  ;       IF SUCCESS, MEMORY DATA STRUCTURES INITIALIZED (IF SO SPECIFIED)
                                  0185  ;       AND MEMORY CONNECTED VIA THE SHARED MEMORY CONTROL BLOCK (SHB).
                                  0185  ;
                                  0185  ;--
                                  0185  SHARE:                                          ; KERNEL ROUTINE
                            0FFC  0185          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; ENTRY MASK
                                  0187  ;
                                  0187  ; MAKE SURE THAT MA780 IS NOT BEING USED FOR MAIN MEMORY.
                                  0187  ;
          50   007C810A 8F   D0   0187          MOVL    #SYSG$_SHMDBLUSE,R0     ; ASSUME ERROR
          54   00000000'GF   D0   018E          MOVL    G^EXE$GL_RPB,R4         ; GET ADDRESS OF RPB
       30 A4   00001800 8F   D3   0195          BITL    #<RPB$M_MPM ! RPB$M_USEMPM>,RPB$L_BOOTR5(R4) ; USED AS MAIN MEM?
                         6F  12   019D          BNEQ    ERR_EXIT                ; BRAND ON ERROR, IS USED AS MAIN MEM
                                  019F  ;
                                  019F  ; FIRST MAKE SURE THAT THE ADAPTER IS INITIALIZED AND CONNECTED
                                  019F  ;
                         54  D4   019F          CLRL    R4                      ; INIT ADAPTER NUMBER
          57   00000000'GF   D0   01A1          MOVL    G^EXE$GL_NUMNEXUS,R7    ; GET ADDRESS OF NUMBER OF NEXUSES
          55   00000000'GF   D0   01A8          MOVL    G^EXE$GL_CONFREGL,R5    ; GET ADDRESS OF CONFREGL ARRAY
             52        6544  D0   01AF  10$:    MOVL    (R5)[R4],R2             ; GET ADAPTER TYPE CODE
                         25  13   01B3          BEQL    30$                     ; BRANCH IF NONE
             40 8F       52  91   01B5          CMPB    R2,#NDT$_MPM0           ; IS ADAPTER A MULTI-PORT MEMORY?
                         1F  1F   01B9          BLSSU   30$                     ; BRANCH IF NOT
             43 8F       52  91   01BB          CMPB    R2,#NDT$_MPM3           ; IS ADAPTER A MULTI-PORT MEMORY?
                         19  1A   01BF          BGTRU   30$                     ; BRANCH IF NOT
          53   00000000'GF   D0   01C1          MOVL    G^IOC$GL_ADPLIST,R3     ; GET ADDRESS OF FIRST ADAPTER BLOCK
             54       0C A3  B1   01C8  20$:    CMPW    ADP$W_TR(R3),R4         ; IS THIS THE BLOCK FOR THE MEMORY?
                         0C  13   01CC          BEQL    30$                     ; BRANCH IF YES - NO NEED TO CREATE ONE
             53       04 A3  D0   01CE          MOVL    ADP$L_LINK(R3),R3       ; GET ADDRESS OF NEXT BLOCK
                         F4  12   01D2          BNEQ    20$                     ; BRANCH IF THERE IS ONE
               00000000'GF   16   01D4          JSB     G^INI$MPMADP            ; ELSE, CREATE AN ADAPTER CONTROL BLOCK
                                  01DA  30$:
             D1 54       57  F2   01DA          AOBLSS  R7,R4,10$               ; INCREMENT ADAPTER NUMBER AND LOOP
                                  01DE  ;
                                  01DE  ; FIND THE SPECIFIED SHARED MEMORY UNIT AND GET ITS ADDRESS
                                  01DE  ;
                                  01DE  FIND_UNIT:
          56   00000000'GF   D0   01DE          MOVL    G^IOC$GL_ADPLIST,R6     ; GET ADDR OF FIRST ADAPTER BLOCK
                         20  13   01E5          BEQL    30$                     ; BRANCH IF NONE
                                  01E7  10$:                                    ; ADAPTER SEARCH LOOP
```

```
      0E A6   00'8F   91  01E7              CMPB    #AT$_MPM,ADP$W_ADPTYPE(R6) ; IS ADAPTER A MULTI-PORT?
                 13   12  01EC              BNEQ    20$                     ; BRANCH IF NOT
           54    66   D0  01EE              MOVL    ADP$L_CSR(R6),R4        ; GET CSR OF ADAPTER
        51    1C A4   D0  01F1              MOVL    MPM$L_MR(R4),R1         ; GET MAINTENENCE VALUE
                 0E   EF  01F5              EXTZV   #MPM$V_MR_UNIT,-        ; GET UNIT NUMBER
      50   51    02       01F7                      #MPM$S_MR_UNIT,R1,R0    ;
      0008'CF    50   B1  01FA              CMPW    R0,SHR_W_UNIT           ; IS IT DESIRED UNIT NUMBER?
                 16   13  01FF              BEQLU   INIT                    ; BRANCH IF YES
                          0201      20$:
        56    04 A6   D0  0201              MOVL    ADP$L_LINK(R6),R6       ; GET ADDR OF NEXT ADAPTER BLOCK
                 E0   12  0205              BNEQ    10$                     ; BRANCH IF ONE EXISTS
                          0207      30$:
 50    007C8042 8F    D0  0207              MOVL    #SYSG$_NOSUCHMEM,R0     ; SET FAILURE
                          020E      ERR_EXIT:
                      04  020E              RET                             ; EXIT
                          020F      ;
                          020F      ; SUBROUTINE USED BY INI$MPMADP TO ALLOCATE NON-PAGED POOL AND EXIT ROUTINE
                          020F      ; CALL IF FAILURE
                          020F      ;
                          020F      INI$ALONONPAGED::                       ;
            FDEE'     30  020F              BSBW    IOGEN$ALLOBLOCK         ; ALLOCATE A BLOCK
            01 50     E9  0212              BLBC    R0,10$                  ; BRANCH IF FAILURE
                      05  0215              RSB                             ; ELSE, OK
                      04  0216      10$:    RET                             ; EXIT ROUTINE WITH STATUS
                          0217      ;
                          0217      ; INITIALIZE AND/OR CONNECT SHARED MEMORY
                          0217      ;
                          0217      INIT:
     003F'CF     56   D0  0217              MOVL    R6,SHR_L_ADP            ; SAVE ADP BLOCK ADDRESS
            0062      30  021C              BSBW    CREATE_SHB              ; CREATE SHARED MEM CONTROL BLCK
            5B 50     E9  021F              BLBC    R0,EXIT                 ; BRANCH IF ERROR
                          0222              SETIPL  #IPL$_HWCLK-1           ; SYNCHRONIZE LOCAL ACCESSORS
            00A4      30  0225              BSBW    MAP_DATAPAGE            ; MAP THE DATAPAGE
            52 50     E9  0228              BLBC    R0,EXIT                 ; BRANCH IF ERROR
            012E      30  022B              BSBW    LOCK_DATAPAGE           ; LOCK THE DATAPAGE
              00      E1  022E              BBC     #SHR_OPT_V_INIT,-       ; BRANCH IF /INIT NOT SPECIFIED
       1A 0026'CF         0230                      SHR_B_OPTIONS,CONNECT   ;
            015E      30  0234              BSBW    CHECK_INIT              ; CHECK IF OK TO INITIALIZE
            14 50     E9  0237              BLBC    R0,CONNECT              ; BRANCH IF NOT
            01A9      30  023A              BSBW    INIT_DATAPAGE           ; INITIALIZE THE DATAPAGE
            3A 50     E9  023D              BLBC    R0,UNLOCK_EXIT          ; BRANCH IF ERROR
            031F      30  0240              BSBW    MAP_STRUCTURES          ; MAP THE OTHER DATA STRUCTURES
            34 50     E9  0243              BLBC    R0,UNLOCK_EXIT          ; BRANCH IF ERROR
            035E      30  0246              BSBW    INIT_STRUCTURES         ; INIT THE OTHER DATA STRUCTURES
            2E 50     E9  0249              BLBC    R0,UNLOCK_EXIT          ; BRANCH IF ERROR
              11      11  024C              BRB     CONNECTED               ; INIT COMPLETED SUCCESSFULLY
                          024E      ;
                          024E      ; JUST CONNECT TO SHARED MEMORY
                          024E      ;
                          024E      CONNECT:                                ; CONNECT TO SHARED MEMORY
              00      E0  024E              BBS     #SHB$V_CONNECT,-        ; BRANCH IF ALREADY CONNECTED
         27 08 A5         0250                      SHB$B_FLAGS(R5),UNLOCK_EXIT
            030C      30  0253              BSBW    MAP_STRUCTURES          ; MAP THE OTHER DATA STRUCTURES
            21 50     E9  0256              BLBC    R0,UNLOCK_EXIT          ; BRANCH IF ERROR
            0474      30  0259              BSBW    CONNECT_MEM             ; CONNECT TO DATA STRUCTURES
            1B 50     E9  025C              BLBC    R0,UNLOCK_EXIT          ; BRANCH IF ERROR
                          025F      CONNECTED:                              ; CONNECTED SUCCESSFULLY
        0B A5    01   88  025F   508            BISB    #SHB$M_CONNECT,SHB$B_FLAGS(R5) ; SET MEMORY CONNECTED
                          0263   509            DSBINT                          ; LOCK OUT INTERRUPTS
     54    1C A5   D0  0269   510            MOVL    SHB$L_ADP(R5),R4        ; GET ADDRESS OF ADP FOR THIS MA780
     55    15 A5   9A  026D   511            MOVZBL  SHB$B_PORT(R5),R5       ; GET OWN PORT NUMBER
    00000000'GF    16  0271   512            JSB     G^MA$REQUEST            ; FORCE INTERRUPT ON OWN PORT TO
                          0277   513            ENBINT                          ;  IMMEDIATELY PROCESS DANGLING PRQS
                          027A   514 UNLOCK_EXIT:                               ; UNLOCK DATAPAGE AND EXIT
             0111     30  027A   515            BSBW    UNLOCK_DATAPAGE         ; UNLOCK DATAPAGE
                          027D   516 EXIT:                                      ; EXIT KERNEL ROUTINE
                          027D   517            SETIPL  #0                      ; RESTORE NORMAL IPL
                    04    0280   518            RET                             ; RETURN
                          0281   519
```

```
                              0281          .SBTTL  CREATE SHARED MEMORY CONTROL BLOCK
                              0281    ;++
                              0281    ;
                              0281    ; CREATE_SHB - CREATE SHARED MEMORY CONTROL BLOCK
                              0281    ;
                              0281    ; THIS ROUTINE IS CALLED TO CREATE A SHARED MEMORY CONTROL BLOCK
                              0281    ; IN THE PROCESSOR'S LOCAL MEMORY POOL.
                              0281    ;
                              0281    ; INPUTS:
                              0281    ;
                              0281    ;       R4 = ADDRESS OF NEXUS CSR
                              0281    ;       R6 = ADAPTER CONTROL BLOCK ADDRESS
                              0281    ;
                              0281    ; OUTPUTS:
                              0281    ;
                              0281    ;       R0 = SUCCESS OR FAILURE STATUS
                              0281    ;       R5 = ADDRESS OF SHARED MEMORY CONTROL BLOCK
                              0281    ;
                              0281    ;       IF SHB FOR MEMORY DID NOT EXIST, IT IS CREATED AND LINKED
                              0281    ;       INTO SHB LIST (EXE$GL_SHBLIST).
                              0281    ;--
                              0281    CREATE_SHB:                                      ; CREATE SHB
                              0281    ;
                              0281    ; CHECK IF SHARED MEMORY CONTROL BLOCK FOR THIS MEMORY ALREADY EXISTS.
                              0281    ;
                              0281          ASSUME  SHB$L_LINK EQ 0
55   00000000'GF      DE      0281          MOVAL   G^EXE$GL_SHBLIST,R5      ; GET ADDR OF SHB LIST
                              0288    10$:
              65      D5      0288          TSTL    SHB$L_LINK(R5)           ; IS THERE A NEXT SHB?
              0C      13      028A          BEQL    20$                      ; BRANCH IF NOT
         55   65      D0      028C          MOVL    SHB$L_LINK(R5),R5        ; GET ADDR OF NEXT SHB
           14 A5      91      028F          CMPB    SHB$B_NEXUS(R5),-        ; IS THIS THE NEXUS?
           0C A6              0292                  ADP$W_TR(R6)             ;
              F2      12      0294          BNEQ    10$                      ; BRANCH IF NOT - TRY NEXT ONE
              30      11      0296          BRB     30$                      ; ELSE - ALREADY EXISTS
                              0298    ;
                              0298    ; CREATE A SHARED MEMORY CONTROL BLOCK FOR THIS MEMORY PORT
                              0298    ;
                              0298    20$:                                             ;
         51   20      9A      0298          MOVZBL  #SHB$K_LENGTH,R1         ; SET SIZE OF SHB
           FD62'      30      029B          BSBW    IOGEN$ALLOBLOCK          ; ALLOCATE THE SHB
           2A 50      E9      029E          BLBC    R0,40$                   ; BRANCH IF FAILURE
         65   52      D0      02A1          MOVL    R2,SHB$L_LINK(R5)        ; SET FORWARD LINK TO SHB
         55   52      D0      02A4          MOVL    R2,R5                    ; SET ADDR OF SHB
                              02A7                                           ; UNITIALIZED FIELDS ARE ZERO!
      08 A5   51      B0      02A7          MOVW    R1,SHB$W_SIZE(R5)        ; SET SIZE OF SHB IN SHB
      0A A5   2A      90      02AB          MOVB    #DYN$C_SHB,SHB$B_TYPE(R5) ; SET TYPE OF SHB IN SHB
      30 A6   55      D0      02AF          MOVL    R5,ADP$L_SHB(R6)         ; SET LINK TO SHB IN ADP
      1C A5   56      D0      02B3          MOVL    R6,SHB$L_ADP(R5)         ; SET LINK TO ADP IN SHB
   14 A5   0C A6      90      02B7          MOVB    ADP$W_TR(R6),SHB$B_NEXUS(R5) ; SET NEXUS NUMBER
         50   64      D0      02BC          MOVL    MPM$L_CSR(R4),R0         ; GET CSR
              00      EF      02BF          EXTZV   #MPM$V_CSR_PORT,-        ; GET PORT NUMBER
    50   50   02              02C1                  #MPM$S_CSR_PORT,R0,R0
      15 A5   50      90      02C4          MOVB    R0,SHB$B_PORT(R5)        ; SET PORT NUMBER
                              02C8    30$:
         50   01      D0      02C8          MOVL    #1,R0                    ; SET SUCCESS
                              02CB    40$:
                                05  02CB   578          RSB                                   ; RETURN
                                    02CC   579
```

```
                           02CC         .SBTTL  MAP THE DATAPAGE
                           02CC  ;++
                           02CC  ;
                           02CC  ; MAP_DATAPAGE - MAP SHARED MEMORY DATAPAGE (LAST PAGE IN MEMORY)
                           02CC  ;
                           02CC  ; THIS ROUTINE IS CALLED TO MAP THE SHARED MEMORY DATAPAGE INTO
                           02CC  ; THE SYSTEM VIRTUAL ADDRESS SPACE.
                           02CC  ;
                           02CC  ; INPUTS:
                           02CC  ;
                           02CC  ;       R4 = ADDR OF NEXUS CSR
                           02CC  ;       R5 = ADDR OF SHARED MEMORY CONTROL BLOCK (SHB)
                           02CC  ;
                           02CC  ;       IPL MUST BE IPL$_SYNCH.
                           02CC  ;
                           02CC  ; OUTPUTS:
                           02CC  ;
                           02CC  ;       R0 = SUCCESS OR FAILURE STATUS.
                           02CC  ;       R6 = ADDR OF DATAPAGE
                           02CC  ;
                           02CC  ;--
                           02CC  MAP_DATAPAGE:                                          ; MAP THE DATAPAGE
         57   0C A4   D0   02CC          MOVL    MPM$L_INV(R4),R7                      ; GET INVALIDATION REG VALUE
            0B   14   EF   02D0          EXTZV   #MPM$V_INV_STADR,#MPM$S_INV_STADR,- ;GET STARTING SBI
            50   57        02D3                  R7,R0                                 ;LONGWORD ADDR<26:16> OF MEMORY
       50   50   09   78   02D5          ASHL    #16+2-VA$V_VPN,R0,R0                  ; CONVERT TO A PFN
       002B'CF   50   D0   02D9          MOVL    R0,SHR_L_MEMPFN                       ; SAVE MEMORY STARTING PFN
            03   10   EF   02DE          EXTZV   #MPM$V_INV_MEMSZ,#MPM$S_INV_MEMSZ,- ; GET MEMORY SIZE IN
            57   57        02E1                  R7,R7                                 ; 256KB BOARD INCREMENTS
                 57   D6   02E3          INCL    R7                                    ; (0 = 1 BOARD)
   57   00000200 8F   C4   02E5          MULL    #<256*1024>/512,R7                    ; CONVERT TO PAGES
       0027'CF   57   D0   02EC          MOVL    R7,SHR_L_MEMSIZE                      ; SAVE MEMORY SIZE
            57   50   C0   02F1          ADDL    R0,R7                                 ; COMPUTE PFN OF LAST PAGE
                 57   D7   02F4          DECL    R7                                    ; ...
                           02F6
         56   04 A5   D0   02F6          MOVL    SHB$L_DATAPAGE(R5),R6                 ; DATAPAGE ALREADY MAPPED?
                 2C   12   02FA          BNEQ    20$                                   ; BRANCH IF YES
                           02FC          ASSUME  SHD$K_LENGTH LE 512                   ; ASSUME 1 PAGE
            51   01   D0   02FC          MOVL    #1,R1                                 ; SET NUMBER PAGES
      00000000'GF   16     02FF          JSB     G^IOC$ALLOSPT                         ; ALLOCATE A SPT ENTRY
            08   50   E8   0305          BLBS    R0,10$                                ; BRANCH IF SUCCESS
   50   007C8022 8F   D0   0308          MOVL    #SYSG$_SPTFULL,R0                     ; SET FAILURE STATUS
                 05        030F          RSB                                           ; EXIT
                           0310  10$:
       56   52   09   78   0310          ASHL    #VA$V_VPN,R2,R6                       ; CONVERT VPN TO VA
   56   80000000 8F   C8   0314          BISL    #VA$M_SYSTEM,R6                       ; ADD SYSTEM SPACE TO VA
         04 A5   56   D0   031B          MOVL    R6,SHB$L_DATAPAGE(R5)                 ;  AND SAVE IN SHB
6342   57   B0000000 8F   C9   031F          BISL3   #<PTE$C_ERKW!PTE$M_VALID>,R7,- ; FILL-IN DATAPAGE SPT
                           0328                  (R3)[R2]                              ; ENTRY AND SET VALID
                           0328  20$:
   50   00000000'GF   D0   0328          MOVL    G^EXE$GL_RPB,R0                       ; GET ADDRESS OF RPB
   50   000000BC 8F   C0   032F          ADDL    #RPB$L_MEMDSC,R0                      ; POINT TO FIRST MEMORY DESCRIPTOR
                           0336  30$:
    14 A5   03 A0   91     0336          CMPB    3(R0),SHB$B_NEXUS(R5)                 ; DOES MA780 TR NUMBER MATCH THIS DSC?
                 16   13   033B          BEQL    40$                                   ; BR IF FOUND THE MEMORY DESCRIPTOR
            50   08   C0   033D          ADDL    #8,R0                                 ; POINT TO NEXT DESCRIPTOR
                 60   D5   0340          TSTL    (R0)                                  ; IS THERE ANOTHER MEMORY TO CHECK?
```

```
                         F2   12  0342           BNEQ    30$                         ; BR IF THERE IS ANOTHER VALID DSC
                                  0344      ;
                                  0344      ; NO MEMORY DESCRIPTOR WAS FOUND FOR THIS MA780.  THEREFORE, IT WAS
                                  0344      ; PROBABLY POWERED UP AFTER THE SYSTEM WAS BOOTED.  A MEMORY DESCRIPTOR
                                  0344      ; IN THE RPB MUST BE CREATED SO THAT THE MA780 PAGES WILL GET WRITTEN
                                  0344      ; TO THE DUMP FILE DURING A BUGCHECK.
                                  0344      ;
             80   0027'CF    D0   0344           MOVL    SHR_L_MEMSIZE,(R0)+         ; SET # OF PAGES OF MEMORY
          FF A0      14 A5   90   0349           MOVB    SHB$B_NEXUS(R5),-1(R0)      ; SET TR # OF MEMORY
             60   002B'CF    D0   034E           MOVL    SHR_L_MEMPFN,(R0)           ; SET STARTING PHYS ADR OF MEMORY
                                  0353      40$:
                  50     01  D0   0353           MOVL    #1,R0                       ; SET SUCCESS
           0037'CF       56  D0   0356           MOVL    R6,SHR_L_DATAPAGE           ; SAVE ADDRESS OF DATAPAGE
                             05   035B           RSB                                 ; RETURN
                                  035C
```

```
                                    035C          .SBTTL  LOCK/UNLOCK THE DATAPAGE
                                    035C   ;++
                                    035C   ;
                                    035C   ; LOCK_DATAPAGE - LOCK THE DATAPAGE FOR INITIALIZATION/CONNECTION
                                    035C   ; UNLOCK_DATAPAGE - UNLOCK THE DATAPAGE
                                    035C   ;
                                    035C   ; THE INIT FLAG IS CLEAR WHEN IT IS LOCKED FOR INITIALIZATION.  THIS IS
                                    035C   ; BECAUSE THE MEMORY IS INITIALIZED TO ALL 1'S WHEN IT IS POWERED ON AND
                                    035C   ; THE COMPLETE TIMEOUT WOULD HAVE TO ELAPSE EVERYTIME A NEWLY POWERED-ON
                                    035C   ; MEMORY WAS INITIALIZED.  TO AVOID THIS, THE SENSE OF THE LOCK IS
                                    035C   ; REVERSED.
                                    035C   ;
                                    035C   ; INPUTS:
                                    035C   ;
                                    035C   ;       R5 = ADDRESS OF SHARED MEMORY CONTROL BLOCK (SHB)
                                    035C   ;       R6 = ADDRESS OF DATAPAGE
                                    035C   ;
                                    035C   ;       IPL LESS THAN IPL$_HWCLK SO TIME CAN BE UPDATED.
                                    035C   ;
                                    035C   ; OUTPUTS:
                                    035C   ;       THE INIT FLAG IS CLEARED OR SET, INDICATING A PORT IS CURRENTLY
                                    035C   ;       INITIALIZING/CONNECTING OR DONE INITIALIZING CONNECTING, RESPECTIVELY.
                                    035C   ;
                                    035C   LOCK_DATAPAGE:                                  ; LOCK DATAPAGE INIT LOCK
         50   00000000'GF      7D   035C          MOVQ    G^EXE$GQ_SYSTIME,R0     ; GET CURRENT SYSTEM TIME
         50   08F0D180 8F      C0   0363          ADDL    #INITLOCK_TIMOUT,R0     ; COMPUTE TIMEOUT TIME
                     51  00    D8   036A          ADWC    #0,R1                   ; ...
          02 009F C6     00    E7   036D   10$:   BBCCI   #SHD$V_INITLCK,SHD$B_FLAGS(R6),20$
                     12        11   0373          BRB     30$
        00000004'GF  51        D1   0375   20$:   CMPL    R1,G^EXE$GQ_SYSTIME+4   ; TIMEOUT?
                     EF        1A   037C          BGTRU   10$                     ; IF GTRU, NO
        00000000'GF  50        D1   037E          CMPL    R0,G^EXE$GQ_SYSTIME     ; TIMEOUT?
                     E6        1A   0385          BGTRU   10$                     ; IF GTRU, NO
                                    0387   30$:
          009D C6    15 A5     90   0387          MOVB    SHB$B_PORT(R5),SHD$B_INITLCK(R6) ; SET LOCKING PORT NUMBER
                               05   038D          RSB                             ;
                                    038E
                                    038E   UNLOCK_DATAPAGE:                        ; UNLOCK THE DATAPAGE INIT LOCK
          00 009F C6     00    E6   038E          BBSSI   #SHD$V_INITLCK,SHD$B_FLAGS(R6),10$
                                    0394   10$:
                               05   0394          RSB
                                    0395
```

```
                          0395         .SBTTL  CHECK IF MEMORY CAN BE INITIALIZED
                          0395  ;++
                          0395  ;
                          0395  ; CHECK_INIT - CHECK IF MEMORY CAN BE INITIALIZED
                          0395  ;
                          0395  ; THIS ROUTINE IS CALLED TO CHECK THAT NO OTHER PORTS ARE USING THE
                          0395  ; THE MEMORY AND IT IS ALRIGHT TO INTIALIZE IT.
                          0395  ;
                          0395  ; INPUTS:
                          0395  ;
                          0395  ;        R4 = ADDR OF NEXUS CSR
                          0395  ;        R5 = ADDR OF SHB
                          0395  ;        R6 = ADDR OF DATAPAGE (SHD)
                          0395  ;
                          0395  ; OUTPUTS:
                          0395  ;
                          0395  ;        R0 = SUCCESS IF MEMORY CAN BE INITIALIZED.
                          0395  ;
                          0395  ;        THIS PORT'S REFERENCE COUNT TO THE MEMORY IS CHECKED, IF IT IS NON-ZERO,
                          0395  ;        THE MEMORY CAN'T BE INITIALIZED.
                          0395  ;
                          0395  ;        THE OTHER PORTS ARE POLLED TO SEE IF THEY ARE CONNECTED TO THE MEMORY
                          0395  ;        BY CLEARING A POLLING MASK AND INTERRUPTING ALL THE PORTS.  IF A PORT
                          0395  ;        IS CONNECTED, IT WILL SET ITS POLLING FLAG, INDICATING THE MEMORY
                          0395  ;        SHOULD NOT BE INITIALIZED.  IF THE TIMEOUT EXPIRES AND NO PORT HAS
                          0395  ;        SET A POLLING FLAG, IT IS OK TO INITIALIZE.
                          0395  ;
                          0395  ;--
                          0395  CHECK_INIT:                                     ; CHECK IF MEMORY CAN BE INITED
          0C A5    D5     0395          TSTL    SHB$L_REFCNT(R5)                ; ANY REFERENCES TO MEMORY?
             49    12     0398          BNEQ    NO_INIT                         ; BRANCH IF YES
                          039A  ;
                          039A  ; POLL OTHER PORTS TO SEE IF THEY ARE CONNECTED TO THE MEMORY
                          039A  ;
                          039A  POLL:
        0070 8F    BB     039A          PUSHR   #^M<R4,R5,R6>                   ; SAVE REGISTERS
        00A6 C6    B4     039E          CLRW    SHD$W_POLL(R6)                  ; CLEAR POLLING FLAGS
     54   1C A5    D0     03A2          MOVL    SHB$L_ADP(R5),R4                ; SET ADDRESS OF ADAPTER CONTROL BLOCK
             55    D4     03A6          CLRL    R5                              ; INIT PORT NUMBER
    00000000'GF    16     03A8  5$:     JSB     G^MA$REQUEST                    ; WAKEUP THE PROCESSOR AT THE PORT
       F6 55  04   F2     03AE          AOBLSS  #MPM$C_PORTS,R5,5$              ; INCREMENT PORT NUMBER AND LOOP
        0070 8F    BA     03B2          POPR    #^M<R4,R5,R6>                   ; RESTORE REGISTERS
                          03B6
 50  00000000'GF   7D     03B6          MOVQ    G^EXE$GQ_SYSTIME,R0             ; GET CURRENT SYSTEM TIME
 50   02FAF080 8F  C0     03BD          ADDL    #INITPOLL_TIMOUT,R0             ; COMPUTE TIMEOUT TIME
         51   00   D8     03C4          ADWC    #0,R1                           ; ...
        00A6 C6    B5     03C7  10$:    TSTW    SHD$W_POLL(R6)                  ; ANY PORT ACTIVE?
             16    12     03CB          BNEQ    NO_INIT                         ; IF NEQ, YES - CAN'T INITIALIZE
 00000004'GF  51   D1     03CD  20$:    CMPL    R1,G^EXE$GQ_SYSTIME+4           ; TIMEOUT?
             F1    1A     03D4          BGTRU   10$                             ; IF GTRU, NO
 00000000'GF  50   D1     03D6          CMPL    R0,G^EXE$GQ_SYSTIME             ; TIMEOUT?
             E8    1A     03DD          BGTRU   10$                             ; IF GTRU, NO
                          03DF  30$:
         50   01   D0     03DF          MOVL    #1,R0                           ; OK TO INITIALIZE
                   05     03E2          RSB                                     ; RETURN
                          03E3
                          03E3  NO_INIT:                                        ; NOT OK TO INITIALIZE
```

```
                                   50  D4  03E3   754          CLRL    R0                          ; SET FAILURE
                                       05  03E5   755          RSB                                 ; RETURN
                                           03E6   756
```

```
                           03E6         .SBTTL  INITIALIZE THE DATAPAGE
                           03E6  ;++
                           03E6  ;
                           03E6  ; INIT_DATAPAGE - INITIALIZE THE DATAPAGE
                           03E6  ;
                           03E6  ; THIS ROUTINE IS CALLED TO INTIALIZE THE SHARED MEMORY DATAPAGE
                           03E6  ; FIELDS AND ALLOCATE THE OTHER DATA STRUCTURES.
                           03E6  ;
                           03E6  ; INPUTS:
                           03E6  ;
                           03E6  ;        R4 = ADDR OF NEXUS CSR
                           03E6  ;        R5 = ADDR OF SHB
                           03E6  ;        R6 = ADDR OF DATAPAGE (SHD)
                           03E6  ;        SHR_VALUES = LIST OF SHARE COMMAND QUALIFIER VALUES
                           03E6  ;
                           03E6  ; OUTPUTS:
                           03E6  ;
                           03E6  ;        R0 = SUCCESS OR FAILURE STATUS.
                           03E6  ;
                           03E6  ;        SHARED MEMORY DATAPAGE IS INITIALIZED.
                           03E6  ;
                           03E6  ;        DURING INITIALIZATION, THE OTHER DATASTRUCTURES ARE ALLOCATED
                           03E6  ;        SO THAT THE APPEAR IN THE FOLLOWING ORDER IN VIRTUAL MEMORY:
                           03E6  ;
                           03E6  ;                +-------------------------+
                           03E6  ;                !        DATAPAGE         !
                           03E6  ;                +-------------------------+
                           03E6  ;                \     PER PORT PAGES      \
                           03E6  ;                +-------------------------+
                           03E6  ;                !         PRQ'S           !
                           03E6  ;                +-------------------------+
                           03E6  ;                !         GSD'S           !
                           03E6  ;                +-------------------------+
                           03E6  ;                !         MBX'S           !
                           03E6  ;                +-------------------------+
                           03E6  ;                !         CEF'S           !
                           03E6  ;                +-------------------------+
                           03E6  ;                !         POOL            !
                           03E6  ;                +-------------------------+
                           03E6  ;                !         BITMAP          !
                           03E6  ;                +-------------------------+
                           03E6  ;                !                         !
                           03E6  ;                \  GLOBAL SECTION PAGES   \
                           03E6  ;                !                         !
                           03E6  ;                +-------------------------+
                           03E6  ;
                           03E6  ;        *** NOTE: THE ORDER OF THESE STRUCTURES IS ASSUMED ****
                           03E6  ;--
                           03E6  INIT_DATAPAGE:                              ; INITIALIZE THE DATAPAGE
                01    90   03E6         MOVB    #SHD$M_INITLCK,-            ; CLEAR THE FLAGS BUT KEEP
           009F C6         03E8                 SHD$B_FLAGS(R6)             ;  THE LOCK SET
          0000'CF     90   03EB         MOVB    SHR_Q_MEMNAME,-             ; SET MEMORY NAME SIZE
             20 A6         03EF                 SHD$T_NAME(R6)              ;
                30    BB   03F1         PUSHR   #^M<R4,R5>                  ; SAVE MOVC REGISTERS
0004'DF   0000'CF     2C   03F3         MOVC5   SHR_Q_MEMNAME,@SHR_Q_MEMNAME+4,- ; SET MEMORY NAME STRING
  21 A6      0F       00   03FA                 #0,#15,SHD$T_NAME+1(R6) ; ZERO-FILLED TO 15 TEXT BYTES
                30    BA   03FE         POPR    #^M<R4,R5>                  ; RESTORE MOVC REGISTERS
           00000000'GF   7D  0400             MOVQ    G^EXE$GQ_SYSTIME,-        ; SET INITIALIZATION TIME
                 30 A6       0406                     SHD$Q_INITTIME(R6)        ;
                    50   D4  0408             CLRL    R0                        ;INIT PORT NUMBER
                             040A             ASSUME  <SHD$Q_PRQWRK & ^B111> EQ 0 ; LIST HEADS MUST BE QUADWORD ALIGNED
              0100 C640  7C  040A     10$:    CLRQ    SHD$Q_PRQWRK(R6)[R0]      ;INIT PORT'S REQUEST WORK QUEUE
              F7 50   04 F2  040F             AOBLSS  #MPM$C_PORTS,R0,10$       ;INCREMENT PORT NUMBER AND LOOP
                00A4 C6  B4  0413             CLRW    SHD$W_PRQWAIT(R6)         ;INIT PRQ WAIT FLAGS
                00E8 C6  B4  0417             CLRW    SHD$W_RESSUM(R6)          ;INIT RESOURCE REPORT SUMMARY FLAGS
                      50 D4  041B             CLRL    R0                        ;SET STARTING RESOURCE NUMBER
              00C8 C640  B4  041D     20$:    CLRW    SHD$W_RESAVAIL(R6)[R0]    ;INIT RESOURCE AVAILABLE FLAGS
              00A8 C640  B4  0422             CLRW    SHD$W_RESWAIT(R6)[R0]     ;INIT RESOURCE WAIT FLAGS
              F2 50   0F F2  0427             AOBLSS  #RSN$_MAX,R0,20$          ;INCREMENT RESOURCE NUMBER AND LOOP
           009C C6    04 90  042B             MOVB    #MPM$C_PORTS,SHD$B_PORTS(R6) ; SET NUMBER OF PORTS
                0022'CF  D0  0430             MOVL    SHR_L_START,-             ; SET RELATIVE PFN
                  14 A6      0434                     SHD$L_GSPFN(R6)           ;  OF 1ST GLOBAL PAGE
                  14 A6  C1  0436             ADDL3   SHD$L_GSPFN(R6),-         ; SET PFN OF 1ST GLOBAL PAGE
        10 A5   002B'CF      0439                     SHR_L_MEMPFN,SHB$L_BASGSPFN(R5)
        18 A6   000A'CF  B0  043E             MOVW    SHR_W_GBLCNT,SHD$W_GSDMAX(R6) ; SET GLOBAL SECTION DESC COUNT
        1A A6   000C'CF  B0  0444             MOVW    SHR_W_MBXCNT,SHD$W_MBXMAX(R6) ; SET MAILBOX COUNT
        1C A6   000E'CF  B0  044A             MOVW    SHR_W_CEFCNT,SHD$W_CEFMAX(R6) ; SET COMMON EVT FLAG CLUST COUNT
             50   15 A5  9A  0450             MOVZBL  SHB$B_PORT(R5),R0         ; GET THIS PORT'S PORT NUMBER
     3C A640    0010'CF  B0  0454             MOVW    SHR_W_GBLQUO,SHD$W_GSDQUOTA(R6)[R0] ; SET THIS PORT'S GSD QUOTA
     5C A640    0012'CF  B0  045B             MOVW    SHR_W_MBXQUO,SHD$W_MBXQUOTA(R6)[R0] ; SET THIS PORT'S MBX QUOTA
     7C A640    0014'CF  B0  0462             MOVW    SHR_W_CEFQUO,SHD$W_CEFQUOTA(R6)[R0] ; SET THIS PORT'S CEF QUOTA
       18 A6    3C A640  B1  0469             CMPW    SHD$W_GSDQUOTA(R6)[R0],SHD$W_GSDMAX(R6) ; IS QUOTA > TABLE SIZE?
                      06 15  046F             BLEQ    30$                       ; BR IF QUOTA IS OK
     3C A640      18 A6  B0  0471             MOVW    SHD$W_GSDMAX(R6),SHD$W_GSDQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
       1A A6    5C A640  B1  0477     30$:    CMPW    SHD$W_MBXQUOTA(R6)[R0],SHD$W_MBXMAX(R6) ; IS QUOTA > TABLE SIZE?
                      06 15  047D             BLEQ    40$                       ; BR IF QUOTA IS OK
     5C A640      1A A6  B0  047F             MOVW    SHD$W_MBXMAX(R6),SHD$W_MBXQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
       1C A6    7C A640  B1  0485     40$:    CMPW    SHD$W_CEFQUOTA(R6)[R0],SHD$W_CEFMAX(R6) ; IS QUOTA > TABLE SIZE?
                      06 15  048B             BLEQ    50$                       ; BR IF QUOTA IS OK
     7C A640      1C A6  B0  048D             MOVW    SHD$W_CEFMAX(R6),SHD$W_CEFQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
                             0493     50$:
                             0493     ;
                             0493     ; FILL-IN DATAPAGE RELATIVE ADDRESSES OF OTHER DATA STRUCTURES AND
                             0493     ; AND KEEP A TOTAL OF THE NUMBER OF PAGES NEEDED FOR THE STUCTURES.
                             0493     ;
         50    01FF 8F   3C  0493             MOVZWL  #511,R0                           ; GET SIZE OF PAGE - 1
         51    0200 8F   3C  0498             MOVZWL  #512,R1                           ; GET SIZE OF A PAGE
                53    01 D0  049D             MOVL    #1,R3                             ; INIT RELATIVE PAGE POINTER
                53    04 C0  04A0             ADDL    #MPM$C_PORTS,R3                   ; RESERVE PER PORT PAGES
     00F0 C6    53    09 78  04A3             ASHL    #VA$V_VPN,R3,-                    ; SET RELATIVE ADDR OF PRQ FREE LIST
                             04A9                     SHD$Q_PRQ(R6)                     ;
          00000040 8F    C5  04A9             MULL3   #PRQ$C_MINLENGTH,-                ; COMPUTE NUMBER BYTES FOR PRQ'S
         52   001E'CF        04AF                     SHR_L_PRQCNT,R2                   ;
                52    50 C0  04B3             ADDL    R0,R2                             ; ROUND-UP TO A PAGE
                52    51 C6  04B6             DIVL    R1,R2                             ; CONVERT TO PAGES
                53    52 C0  04B9             ADDL    R2,R3                             ; COMPUTE RELATIVE PAGE OF GSD'S
       04 A6    53    09 78  04BC             ASHL    #VA$V_VPN,R3,-                    ; SET RELATIVE ADDR OF GSD TABLE
                             04C1                     SHD$L_GSDPTR(R6)                  ;
         52   000A'CF    3C  04C1             MOVZWL  SHR_W_GBLCNT,R2                   ; GET NUMBER OF GSD'S
          00000074 8F    C1  04C6             ADDL3   #GSD$K_SHMGSDLNG,-                ; COMPUTE SIZE OF GSD'S
        002F'CF   10         04CC                     #<MPM$C_PORTS*4>,SHR_L_GSDSIZE ; (LONGWORD REFCNT/PORT)
         52   002F'CF    C4  04D0             MULL    SHR_L_GSDSIZE,R2                  ; COMPUTE NUMBER BYTES NEEDED
                52    50 C0  04D5             ADDL    R0,R2                             ; ROUND-UP TO A PAGE
                52    51 C6  04D8             DIVL    R1,R2                             ; CONVERT TO PAGES
             53     52   C0  04DB            ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF MBX'S
        66   53     09   78  04DE            ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF MBX TABLE
                             04E2                    SHD$L_MBXPTR(R6)                ;
        52   000C'CF     3C  04E2            MOVZWL  SHR_W_MBXCNT,R2                 ; GET NUMBER OF MAILBOXES
             52     30   C4  04E7            MULL    #MBX$K_LENGTH,R2                ; COMPUTE NUMBER BYTES NEEDED
             52     50   C0  04EA            ADDL    R0,R2                           ; ROUND-UP TO A PAGE
             52     51   C6  04ED            DIVL    R1,R2                           ; CONVERT TO PAGES
             53     52   C0  04F0            ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF CEF TABLE
     08 A6   53     09   78  04F3            ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF CEF TABLE
                             04F8                    SHD$L_CEFPTR(R6)                ;
        52   000E'CF     3C  04F8            MOVZWL  SHR_W_CEFCNT,R2                 ; GET NUMBER OF COM EVT FLAG BLOCKS
                    38   C1  04FD            ADDL3   #CEB$C_LENGTH,-                 ; COMPUTE SIZE OF SHMCEB
      0033'CF       18       04FF                    #<MPM$C_PORTS*6>,SHR_L_CEFSIZE ; (WORD REFCNT+SLAVE VA/PORT)
        52   0033'CF     C4  0503            MULL    SHR_L_CEFSIZE,R2                ; COMPUTE NUMBER BYTES NEEDED
             52     50   C0  0508            ADDL    R0,R2                           ; ROUND-UP TO A PAGE
             52     51   C6  050B            DIVL    R1,R2                           ; CONVERT TO PAGES
             53     52   C0  050E            ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF POOL
   00F8 C6   53     09   78  0511            ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF POOL
                             0517                    SHD$Q_POOL(R6)                  ;
             0016'CF     C5  0517            MULL3   SHR_L_POOLBCNT,-                ; COMPUTE SIZE OF POOL IN BYTES
        52   001A'CF         051B                    SHR_L_POOLBSIZ,R2               ;
             52     50   C0  051F            ADDL    R0,R2                           ; ROUND-UP TO A PAGE
             52     51   C6  0522            DIVL    R1,R2                           ; CONVERT TO PAGES
             53     52   C0  0525            ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF BITMAP
     0C A6   53     09   78  0528            ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF BITMAP
                             052D                    SHD$L_GSBITMAP(R6)              ;
                             052D      ;
                             052D      ; COMPUTE NUMBER PAGES LEFT FOR GLOBAL SECTIONS AND SIZE OF BITMAP
                             052D      ; TO REPRESENT GLOBAL SECTION PAGES.
                             052D      ;
               14 A6     C3  052D            SUBL3   SHD$L_GSPFN(R6),-               ; GET MEMORY SIZE
        50   0027'CF         0530                    SHR_L_MEMSIZE,R0                ;  LESS RESERVED AREA
             50     53   C2  0534            SUBL    R3,R0                           ; COMPUTE NUMBER GLOBAL SECTION
                             0537                                                    ; PAGES THAT WILL BE AVAILABLE
                    21   15  0537            BLEQ    100$                            ; BRANCH IF THERE ARE NONE
  51    50   00000FFF 8F C1  0539            ADDL3   #<512*8>-1,R0,R1                ; ROUND-UP TO NUMBER PER
                             0541                                                    ;  PAGE OF BITMAP
        51   00001000 8F C6  0541            DIVL    #<512*8>,R1                     ; COMPUTE NUMBER PAGES FOR BITMAP
     10 A6   50     51   C3  0548            SUBL3   R1,R0,SHD$L_GSPAGCNT(R6)        ; SET NUMBER GLOBAL PAGES AVAIL
                             054D                                                    ;  LESS NUMBER BITMAP PAGES
                    0B   15  054D            BLEQ    100$                            ; BRANCH IF NONE
                  02E2   30  054F            BSBW    DATAPAGE_CRC                    ; COMPUTE DATAPAGE CRC
          38 A6     50   D0  0552            MOVL    R0,SHD$L_CRC(R6)                ; SET CRC
             50     01   D0  0556            MOVL    #1,R0                           ; SET SUCCESS
                         05  0559            RSB                                     ; RETURN
                             055A
                             055A      100$:
        50   007C804A 8F D0  055A            MOVL    #SYSG$_BADPARAM,R0              ; SET FAILURE
                         05  0561            RSB                                     ; RETURN
                             0562
```

```
                              0562          .SBTTL  MAP THE OTHER DATA STRUCTURES
                              0562  ;++
                              0562  ;
                              0562  ; MAP_STRUCTURES - MAP THE OTHER DATA STRUCTURES
                              0562  ;
                              0562  ; THIS ROUTINE IS CALLED TO MAP THE OTHER DATA STRUCTURES INTO
                              0562  ; SYSTEM VIRTUAL ADDRESS SPACE.
                              0562  ;
                              0562  ; INPUTS:
                              0562  ;
                              0562  ;       R4 = ADDR OF CSR NEXUS
                              0562  ;       R5 = ADDR OF SHB
                              0562  ;       R6 = ADDR OF DATAPAGE (SHD)
                              0562  ;
                              0562  ; OUTPUTS:
                              0562  ;
                              0462  ;       R0 = SUCCESS OR FAILURE STATUS.
                              0562  ;
                              0562  ;       THE OTHER SHARED MEMORY DATA STRUCTURES (POOL, MAILBOXES,
                              0562  ;       GLOBAL SECTION DESCRIPTORS, GLOBAL SECTION BITMAP) ARE
                              0562  ;       MAPPED IN SYSTEM SPACE.  THE RELATIVE ADDRESSES IN THE
                              0562  ;       DATAPAGE CAN NOW BE USED TO ACCESS THE STRUCTURES.
                              0562  ;
                              0562  ;       NOTE: THE PAGES ARE MAPPED SO THAT THE HIGHEST NUMBERED PFN
                              0562  ;       HAS THE LOWEST VIRTUAL ADDRESS.
                              0562  ;
                              0562  ;--
                              0562  MAP_STRUCTURES:                                 ; MAP THE OTHER STRUCTURES
              F7 8F     78    0562          ASHL    #-VA$V_VPN,-                    ; GET NUMBER OF PAGES TO MAP
           51    0C A6        0565                  SHD$L_GSBITMAP(R6),R1           ; (BITMAP IS LAST STRUCTURE)
         00000000'GF    16    0568          JSB     G^IOC$ALLOSPT                   ; ALLOCATE SYS PAGE TABLE ENTRIES
                 08 50  E8    056E          BLBS    R0,10$                          ; BRANCH IF SUCCESS
     50  007C8022 8F    D0    0571          MOVL    #SYSG$_SPTFULL,R0               ; SET FAILURE STATUS
                        05    0578          RSB                                     ; RETURN
                              0579  10$:
     50   56    15   09 EF    0579          EXTZV   #VA$V_VPN,#VA$S_VPN,R6,R0 ; GET VPN OF DATAPAGE
       003B'CF    6340  DE    057E          MOVAL   (R3)[R0],SHR_L_SHDPTE           ; SAVE ADDR OF DATAPAGE PTE
              50  6340  D0    0584          MOVL    (R3)[R0],R0                     ; GET PTE OF DATAPAGE (LAST PAGE)
     50   50    15   00 EF    0588          EXTZV   #PTE$V_PFN,#PTE$S_PFN,R0,R0 ; GET PFN OF DATAPAGE
                              058D  20$:                                    ; MAPPING LOOP
                    50  D7    058D          DECL    R0                              ; DECREMENT PFN
         B0000000 8F    C9    058F          BISL3   #<PTE$C_ERKW!PTE$M_VALID>,- ; SET PTE VALID, KERNEL WRITEABLE
            6342    50        0595                  R0,(R3)[R2]                     ;  AND ENTER PFN
                    52  D6    0598          INCL    R2                              ; INCREMENT VPN
                 F0 51  F5    059A          SOBGTR  R1,20$                          ; DECREMENT PAGE COUNT AND LOOP
                              059D
  18 A5    56    0C A6  C1    059D          ADDL3   SHD$L_GSBITMAP(R6),R6,- ; SET ADDR OF END OF POOL
                              05A3                  SHB$L_POOLEND(R5)               ;  (BITMAP ASSUMED TO FOLLOW POOL)
              50    01  D0    05A3          MOVL    #1,R0                           ; SET SUCCESS
                        05    05A6          RSB                                     ; RETURN
                              05A7
```

```
                                05A7          .SBTTL  INITIALIZE THE OTHER DATA STRUCTURES
                                05A7    ;++
                                05A7    ;
                                05A7    ; INIT_STRUCTURES - INITIALIZE THE OTHER DATA STRUCTURES
                                05A7    ;
                                05A7    ; THIS ROUTINE IS CALLED TO INTIALIZE THE OTHER DATA STRUCTURES IN
                                05A7    ; THE SHARED MEMORY.
                                05A7    ;
                                05A7    ; INPUTS:
                                05A7    ;
                                05A7    ;       R4 = ADDR OF NEXUS CSR
                                05A7    ;       R5 = ADDR OF SHB
                                05A7    ;       R6 = ADDR OF DATAPAGE (SHD)
                                05A7    ;
                                05A7    ; OUTPUTS:
                                05A7    ;
                                05A7    ;       THE OTHER SHARED MEMORY DATA STRUCTURES (POOL, MAILBOXES,
                                05A7    ;       GLOBAL SECTION DESCRIPTORS, GLOBAL SECTION BITMAP) ARE
                                05A7    ;       INITIALIZED FOR USE.
                                05A7    ;--
                                05A7    INIT_STRUCTURES:                                ; INITIALIZE THE STRUCTURES
                                05A7    ;
                                05A7    ; INITIALIZE THE GLOBAL SECTION DESCRIPTOR TABLE
                                05A7    ;
         57    56    04 A6   C1 05A7            ADDL3   SHD$L_GSDPTR(R6),R6,R7  ; GET ADDR OF 1ST GSD
               58    18 A6   3C 05AC            MOVZWL  SHD$W_GSDMAX(R6),R8     ; GET COUNT OF GSD'S
                                05B0    10$:                                    ; INIT LOOP
                                05B0            ASSUME GSD$L_GSDBL EQ <GSD$L_GSDFL + 4>
                         67  7C 05B0            CLRQ    GSD$L_GSDFL(R7)         ; INIT FORWARD/BACKWARD LINKS
                   009C C6   90 05B2            MOVB    SHD$B_PORTS(R6),-       ; SET # OF PORTS
                     51 A7      05B6                    GSD$B_PROCCNT(R7)       ;
                        29   9B 05B8            MOVZBW  #DYN$C_SHMGSD,-         ; SET STRUCTURE TYPE AND
                     0A A7      05BA                    GSD$B_TYPE(R7)          ;   CLEAR UNUSED BYTE
                   002F'CF   B0 05BC            MOVW    SHR_L_GSDSIZE,-         ; SET STRUCTURE SIZE
                     08 A7      05C0                    GSD$W_SIZE(R7)          ;
                     50 A7   94 05C2            CLRB    GSD$B_LOCK(R7)          ; CLEAR LOCK
                     53 A7   94 05C5            CLRB    GSD$B_DELETPORT(R7)     ; CLEAR ANY DELETES PENDING
               50    51 A7   9A 05C8            MOVZBL  GSD$B_PROCCNT(R7),R0    ; GET # OF PROCESSOR REF CNTS
               51    74 A7   DE 05CC            MOVAL   GSD$L_PTECNT1(R7),R1    ; GET ADR OF FIRST REF CNT
                        81   D4 05D0    20$:    CLRL    (R1)+                   ; CLEAR ONE REF COUNT
                     FB 50   F5 05D2            SOBGTR  R0,20$                  ; REPEAT FOR EACH PROCESSOR REF COUNT
            57     002F'CF   C0 05D5            ADDL    SHR_L_GSDSIZE,R7        ; INCREMENT GSD POINTER
                     D3 58   F5 05DA            SOBGTR  R8,10$                  ; DECREMENT GSD COUNTER AND LOOP
                                05DD    ;
                                05DD    ; INITIALIZE THE GLOBAL SECTION BITMAP BY PERFORMING A GROSS READ/WRITE
                                05DD    ; TEST ON EACH PAGE.  IF IT PASSES THE TEST, IT IS MARKED AVAILABLE
                                05DD    ; IN THE BITMAP.  IF IT DOESN'T PASS THE  TEST, IT IS MARKED UNAVAILABLE.
                                05DD    ;
                                05DD    ; *** NOTE: TO MAP THE PAGES, THE DATAPAGE IS UNMAPPED ***
                                05DD    ;
                                05DD    INIT_BITMAP:                                    ; INITIALIZE THE BITMAP
                        54   DD 05DD            PUSHL   R4                      ; SAVE REGISTER
         57    56    0C A6   C1 05DF            ADDL3   SHD$L_GSBITMAP(R6),R6,R7 ; GET ADDR OF BITMAP
                        3F   BB 05E4            PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; SAVE REGISTERS DESTROYED BY MOVC
67   0200 8F    00    66 00  2C 05E6            MOVC5   #0,(R6),#0,#512,(R7)    ; ZERO-FILL THE BITMAP PAGE
                        3F   BA 05EE            POPR    #^M<R0,R1,R2,R3,R4,R5>  ; RESTORE REGISTERS CLOBBERED BY MOVC
            58     003B'CF   D0 05F0            MOVL    SHR_L_SHDPTE,R8         ; GET ADDR OF DATAPAGE PTE
```

```
                68      DD  05F5            PUSHL   (R8)                            ; SAVE DATAPAGE PTE
       59    10 A6      D0  05F7            MOVL    SHD$L_GSPAGCNT(R6),R9           ; GET NUMBER OF GLOBAL PAGES
       5A    10 A5      D0  05FB            MOVL    SHB$L_BASGSPFN(R5),R10          ; GET PFN OF 1ST GLOBAL PAGE
                5B      D4  05FF            CLRL    R11                             ; INIT CURRENT RELATIVE PAGE NUMBER
                            0601    10$:
  68   15    00 5A      F0  0601            INSV    R10,#PTE$V_PFN,#PTE$S_PFN,(R8) ; MAP THE PAGE TO TEST
             53 56      D0  0606            MOVL    R6,R3                           ; GET A COPY OF VIRTUAL ADDRESS OF PAGE
       54    5A 09      78  0609            ASHL    #VA$V_VPN,R10,R4                ; COMPUTE PHYSICAL ADDRESS OF PAGE
                            060D            INVALID R3                              ; INVALIDATE VIRTUAL ADDRESS TRANSLATION
             F9ED'      30  0610            BSBW    IOGEN$TEST_MEM                  ; TEST THE PAGE
             06 50      E9  0613            BLBC    R0,20$                          ; BR IF BAD PAGE
       06 67    5B      E2  0616            BBSS    R11,(R7),30$                    ; SET PAGE OK
                04      11  061A            BRB     30$                             ;
                            061C    20$:
       00 67    5B      E5  061C            BBCC    R11,(R7),30$                    ; SET PAGE BAD
                            0620    30$:
                5A      D6  0620            INCL    R10                             ; INCREMENT PFN
       DB 5B    59      F2  0622            AOBLSS  R9,R11,10$                      ; INCREMENT CURRENT PAGE NUMBER AND LOOP
             68 8ED0        0626            POPL    (R8)                            ; RESTORE DATAPAGE PTE (REMAP)
                            0629            INVALID R6                              ; INVALIDATE VIRTUAL ADDRESS TRANSLATION
             54 8ED0        062C            POPL    R4                              ; RESTORE REGISTER
      00000000'GF       16  062F            JSB     G^MA$INITIAL                    ; CLEAR ANY PORT ERRORS
                            0635    ;
                            0635    ; INITIALIZE THE POOL
                            0635    ;
                            0635    INIT_POOL:                                      ; INITIALIZE THE POOL
  50   56   00F8 C6     C1  0635            ADDL3   SHD$Q_POOL(R6),R6,R0            ; GET ADDR OF FIRST B OCK
            00F8 C6     7C  063B            CLRQ    SHD$Q_POOL(R6)                  ; SET QUEUE EMPTY
       51   0016'CF     D0  063F            MOVL    SHR_L_POOLBCNT,R1               ; GET NUMBER OF BLOCKS
       52   001A'CF     D0  0644            MOVL    SHR_L_POOLBSIZ,R2               ; GET BLOCK SIZE
                            0649    10$:
     00F8 C6    60      5C  0649            INSQHI  (R0),SHD$Q_POOL(R6)             ; INSERT BLOCK IN LIST
       08 A0    52      B0  064E            MOVW    R2,ACB$W_SIZE(R0)               ; SET SIZE OF BLOCK IN BLOCK
          50    52      C0  0652            ADDL    R2,R0                           ; INCREMENT BLOCK POINTER
             F1 51      F5  0655            SOBGTR  R1,10$                          ; DECREMENT BLOCK COUNT AND LOOP
                            0658    ;
                            0658    ; INTIALIZE THE FREE INTER-PROCESSOR REQUEST BLOCK QUEUE
                            0658    ;
                            0658    INIT_PRQ:                                       ; INITIALIZE FREE PRQ QUEUE
  50   56   00F0 C6     C1  0658            ADDL3   SHD$Q_PRQ(R6),R6,R0             ; GET ADDR OF FIRST BLOCK
            00F0 C6     7C  065E            CLRQ    SHD$Q_PRQ(R6)                   ; SET QUEUE EMPTY
       51   001E'CF     D0  0662            MOVL    SHR_L_PRQCNT,R1                 ; GET NUMBER OF BLOCKS
                            0667    10$:
     00F0 C6    60      5D  0667            INSQTI  (R0),SHD$Q_PRQ(R6)              ; INSERT BLOCK IN LIST
  50    00000040 8F     C0  066C            ADDL    #PRQ$C_MINLENGTH,R0             ; INCREMENT BLOCK POINTER
             F1 51      F5  0673            SOBGTR  R1,10$                          ; DECREMENT BLOCK COUNT AND LOOP
                            0676
                            0676    ;
                            0676    ; INITIALIZE THE MAILBOX TABLE
                            0676    ;
                            0676    INIT_MAILBOXES:                                 ; INITIALIZE THE MAILBOXES
       57   56  66      C1  0676            ADDL3   SHD$L_MBXPTR(R6),R6,R7          ; GET ADDR OF 1ST MAILBOX
          58    1A A6   3C  067A            MOVZWL  SHD$W_MBXMAX(R6),R8             ; GET NUMBER TO INIT
                50      D4  067E            CLRL    R0                              ; INIT INITIALIZED COUNT
                            0680    10$:
             08 A7      94  0680            CLRB    MBX$B_FLAGS(R7)                 ; CLEAR ALL FLAGS
   0A A7    50  01      A1  0683            ADDW3   #1,R0,MBX$W_UNIT(R7)            ; SET UNIT NUMBER
```

```
                          0688                                              ; (FROM 1 TO N, AS 0 IS RESERVED)
             57    30  C0 0688              ADDL    #MBX$K_LENGTH,R7        ; INCREMENT MAILBOX POINTER
          F1 50    58  F2 068B              AOBLSS  R8,R0,10$               ; INCREMENT COUNT AND LOOP
                          068F
                          068F       ;
                          068F       ; INITIALIZE THE COMMON EVENT FLAG TABLE
                          068F       ;
                          068F       INIT_CEF:
    57       56  08 A6 C1 068F              ADDL3   SHD$L_CEFPTR(R6),R6,R7  ; GET ADR OF 1ST SHMCEB IN TABLE
             58  1C A6 3C 0694              MOVZWL  SHD$W_CEFMAX(R6),R8     ; GET NUMBER OF ENTRIES TO INIT
                          0698       10$:
                          0698              ASSUME CEB$L_CEBBL EQ <CEB$L_CEBFL+4>
                    67 7C 0698              CLRQ    CEB$L_CEBFL(R7)         ; INIT FLAGS
    1D A7     009C C6  90 069A              MOVB    SHD$B_PORTS(R6),CEB$B_PROCCNT(R7) ; SET # OF PROCESSORS
       0A A7       2E  9B 06A0              MOVZBW  #DYN$C_SHMCEB,CEB$B_TYPE(R7) ; SET TYPE OF DATA STRUCTURE
    08 A7    0033'CF   B0 06A4              MOVW    SHR_L_CEFSIZE,CEB$W_SIZE(R7) ; SET SIZE OF SHMCEB
                 1C A7 94 06AA              CLRB    CEB$B_LOCK(R7)          ; CLEAR OWNER OF CEB LOCK
                 1F A7 94 06AD              CLRB    CEB$B_DELETPORT(R7)     ; CLEAR DELETOR OF CEB
             50  1D A7 9A 06B0              MOVZBL  CEB$B_PROCCNT(R7),R0    ; GET # OF PROCESSORS MAX
          51  38 A740  DE 06B4              MOVAL   CEB$L_VASLAVE1(R7)[R0],R1 ; GET ADR OF FIRST PROC REF COUNT
             52  38 A7 DE 06B9              MOVAL   CEB$L_VASLAVE1(R7),R2   ; GET ADR OF FIRST SLAVE CEB VA
                    82 D4 06BD       20$:   CLRL    (R2)+                   ; CLEAR THE VA OF SLAVE CEB FOR PROC
                    81 B4 06BF              CLRW    (R1)+                   ; CLEAR REF COUNT FOR THIS PROCESSOR
                 F9 50 F5 06C1              SOBGTR  R0,20$                  ; REPEAT FOR EACH PROCESSOR
          57 0033'CF   C0 06C4              ADDL    SHR_L_CEFSIZE,R7        ; GET NEXT SHMCEB IN TABLE
                 CC 58 F5 06C9              SOBGTR  R8,10$                  ; INIT EACH SHMCEB IN TABLE
                          06CC
             50    01  D0 06CC              MOVL    #1,R0                   ; SET SUCCESS
                       05 06CF              RSB
                          06D0
```

```
                                06D0           .SBTTL  CONNECT TO OTHER DATA STRUCTURES
                                06D0     ;++
                                06D0     ;
                                06D0     ; CONNECT_MEM - CONNECT TO OTHER SHARED MEMORY DATA STRUCTURES
                                06D0     ;
                                06D0     ; THIS ROUTINE IS CALLED TO JUST CONNECT THIS PORT TO AN ALREADY
                                06D0     ; INTIALIZED SHARED MEMORY.
                                06D0     ;
                                06D0     ; INPUTS:
                                06D0     ;
                                06D0     ;         R4 = ADDR OF NEXUS CSR
                                06D0     ;         R5 = ADDR OF SHB
                                06D0     ;         R6 = ADDR OF DATAPAGE (SHD)
                                06D0     ;
                                06D0     ; OUTPUTS:
                                06D0     ;
                                06D0     ;         R0 = SUCCESS OR FAILURE STATUS.
                                06D0     ;
                                06D0     ;         THE DATAPAGE IS FIRST TESTED TO BE SURE THAT IT IS INITIALIZED.
                                06D0     ;         IF IT HAS BEEN INITIALIZED, THEN THE EXISTING DATA STRUCTURES
                                06D0     ;         (MAILBOXES, GLOBAL SECTION DESCRIPTORS) ARE SCANNED FOR
                                06D0     ;         ONES THAT ARE MARKED FOR DELETE.  IF THE STRUCTURE HAD
                                06D0     ;         REFERENCES ONLY FROM THIS PORT, THEN THE STRUCTURE IS DELETED.
                                06D0     ;
                                06D0     ;--
                                06D0     CONNECT_MEM:                               ; CONNECT DATA STRUCTURES
                  0161     30   06D0             BSBW    DATAPAGE_CRC               ; COMPUTE DATAPAGE CRC
            38 A6   50     D1   06D3             CMPL    R0,SHD$L_CRC(R6)           ; CRC COMPARE?
                    08     13   06D7             BEQL    10$                        ; BRANCH IF YES
     50   007C8052 8F      D0   06D9             MOVL    #SYSG$_BADCHKSUM,R0        ; SET FAILURE
                           05   06E0             RSB                                ; RETURN
                                06E1     10$:
            50   20 A6     9A   06E1             MOVZBL  SHD$T_NAME(R6),R0          ; GET SIZE OF MEMORY NAME
            21 A6   50     2D   06E5             CMPC5   R0,SHD$T_NAME+1(R6),-      ; IS NAME THE ONE SPECIFIED?
 0004'DF    0000'CF  00         06E9                     #0,SHR_Q_MEMNAME,@SHR_Q_MEMNAME+4
                    08     13   06F0             BEQLU   20$                        ; BRANCH IF YES
     50   007C805A 8F      D0   06F2             MOVL    #SYSG$_INCMEMNAM,R0        ; SET FAILURE
                           05   06F9             RSB                                ; RETURN
                                06FA
                                06FA     20$:
                 14 A6     C1   06FA             ADDL3   SHD$L_GSPFN(R6),-          ; SET PFN OF 1ST GLOBAL PAGE
        10 A5   002B'CF         06FD                     SHR_L_MEMPFN,SHB$L_BASGSPFN(R5)
      000C'CF     1A A6    B0   0702             MOVW    SHD$W_MBXMAX(R6),SHR_W_MBXCNT ; SAVE NUMBER OF MAILBOXES
           50     15 A5    9A   0708             MOVZBL  SHB$B_PORT(R5),R0          ; GET THIS PORT'S PORT NUMBER
     3C A640    0010'CF    B0   070C             MOVW    SHR_W_GBLQUO,SHD$W_GSDQUOTA(R6)[R0] ; SET THIS PORT'S GSD QUOTA
     5C A640    0012'CF    B0   0713             MOVW    SHR_W_MBXQUO,SHD$W_MBXQUOTA(R6)[R0] ; SET THIS PORT'S MBX QUOTA
    7C A640     0014'CF    B0   071A             MOVW    SHR_W_CEFQUO,SHD$W_CEFQUOTA(R6)[R0] ; SET THIS PORT'S CEF QUOTA
      18 A6     3C A640    B1   0721             CMPW    SHD$W_GSDQUOTA(R6)[R0],SHD$W_GSDMAX(R6) ; IS QUOTA > TABLE SIZE?
                    06     15   0727             BLEQ    30$                        ; BR IF QUOTA IS OK
     3C A640      18 A6    B0   0729             MOVW    SHD$W_GSDMAX(R6),SHD$W_GSDQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
      1A A6     5C A640    B1   072F     30$:    CMPW    SHD$W_MBXQUOTA(R6)[R0],SHD$W_MBXMAX(R6) ; IS QUOTA > TABLE SIZE?
                    06     15   0735             BLEQ    40$                        ; BR IF QUOTA IS OK
     5C A640      1A A6    B0   0737             MOVW    SHD$W_MBXMAX(R6),SHD$W_MBXQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
      1C A6     7C A640    B1   073D     40$:    CMPW    SHD$W_CEFQUOTA(R6)[R0],SHD$W_CEFMAX(R6) ; IS QUOTA > TABLE SIZE?
                    06     15   0743             BLEQ    50$                        ; BR IF QUOTA IS OK
     7C A640      1C A6    B0   0745             MOVW    SHD$W_CEFMAX(R6),SHD$W_CEFQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
                                074B     50$:
```

```
                          074B  ;
                          074B  ; RE-INITIALIZE THE REFERENCE COUNTS FOR THIS PORT IN THE GSD TABLE.
                          074B  ; ALSO, IF THIS PORT CREATED ANY OF THE SECTIONS, SET THE CREATOR TO -1
                          074B  ; TO PROHIBIT USE OF NON-EXISTANT SECTION TABLE.  THIS DOES NOT ATTEMPT
                          074B  ; TO RELEASE ANY GLOBAL SECTIONS NOT IN USE BY OTHER PORTS OR GLOBAL SECTIONS
                          074B  ; WHICH WERE ONLY PARTIALLY CREATED BY THIS PORT.
                          074B  ;
                 0F   BB  074B          PUSHR   #^M<R0,R1,R2,R3>        ; SAVE REGISTERS
    50    56  04 A6   C1  074D          ADDL3   SHD$L_GSDPTR(R6),R6,R0  ; GET ADR OF FIRST GSD IN TABLE
          51  18 A6   3C  0752          MOVZWL  SHD$W_GSDMAX(R6),R1     ; GET COUNT OF GSD'S IN TABLE
                 23   15  0756          BLEQ    190$                    ; BR IF NO TABLE TO INIT
          52  15 A5   9A  0758          MOVZBL  SHB$B_PORT(R5),R2       ; GET PORT # FOR THIS PROCESSOR
          53  08 A0   3C  075C          MOVZWL  GSD$W_SIZE(R0),R3       ; GET SIZE OF ONE GSD IN BYTES
            74 A042   D4  0760  110$:   CLRL    GSD$L_PTECNT1(R0)[R2]   ; INITIALIZE THE REF CNT FOR THIS PORT
       0D 60     00   E1  0764          BBC     #GSD$V_VALID,GSD$L_GSDFL(R0),120$ ; BR IF SECTION NOT IN USE
       52 A0     52   91  0768          CMPB    R2,GSD$B_CREATPORT(R0)  ; DID THIS PORT CREATE THE SECTION?
                 07   12  076C          BNEQ    120$                    ; BR IF IT IS NOT THE CREATOR
       52 A0     00   92  076E          MCOMB   #0,GSD$B_CREATPORT(R0)  ; MAKE THIS NOT THE CREATOR
          16 A0       B4  0772          CLRW    GSD$W_GSTX(R0)          ; SET NO SECTION TABLE ENTRY
          50     53   C0  0775  120$:   ADDL2   R3,R0                   ; GET ADR OF NEXT GSD
             E5  51   F5  0778          SOBGTR  R1,110$                 ; REPEAT FOR EACH GSD IN TABLE
                 0F   BA  077B  190$:   POPR    #^M<R0,R1,R2,R3>        ; RESTORE REGISTERS
                          077D
                          077D  ;
                          077D  ; RE-INITIALIZE THE REFERENCE FLAGS FOR THIS PORT IN THE MAILBOXES.
                          077D  ; THIS DOES NOT RELEASE ANY MAILBOXES THAT ARE NOT IN USE BY OTHER PORTS.
                          077D  ;
                          077D  200$:
                          077D          LOCK    #SHD$V_MBXLCK,SHD$B_FLAGS(R6) ; LOCK SHM MAILBOX TABLE
     50    56    66   C1  079B          ADDL3   SHD$L_MBXPTR(R6),R6,R0  ; GET ADDR OF FIRST MAILBOX
       51     1A A6   3C  079F          MOVZWL  SHD$W_MBXMAX(R6),R1     ; GET COUNT OF MAILBOXES IN TABLE
       52     15 A5   9A  07A3          MOVZBL  SHB$B_PORT(R5),R2       ; GET PORT # FOR THIS PROCESSOR
                          07A7  210$:
    00 0C A0     52   E7  07A7          BBCCI   R2,MBX$W_REF(R0),220$   ; CLEAR PORT FLAG
                          07AC  220$:
       09 A0     52   91  07AC          CMPB    R2,MBX$B_CREATPORT(R0)  ; IS THIS MBX OWNED BY THIS PORT?
                 04   12  07B0          BNEQ    230$                    ; BR IF OWNED BY SOME OTHER PORT
          5C A642     B7  07B2          DECW    SHD$W_MBXQUOTA(R6)[R2]  ; SUBTRACT ONE FOR THIS MBX OWNERSHIP
                          07B6  230$:
          50     30   C0  07B6          ADDL    #MBX$K_LENGTH,R0        ; INCREMENT MAILBOX POINTER
             EB  51   F5  07B9          SOBGTR  R1,210$                 ; DECREMENT COUNT AND LOOP
                          07BC          UNLOCK  #SHD$V_MBXLCK,SHD$B_FLAGS(R6) ; UNLOCK SHM MAILBOX TABLE
                          07C5  ;
                          07C5  ; RE-INITIALIZE THE REFERENCE COUNTS FOR THIS PORT IN THE CEF TABLE.
                          07C5  ; THIS RELEASES ANY TEMPORARY COMMON EVENT FLAG CLUSTERS THAT ARE NOT
                          07C5  ; IN USE BY OTHER PORTS.
                          07C5  ;
                 3F   BB  07C5  300$:   PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ;SAVE REGISTERS
    50    56  08 A6   C1  07C7          ADDL3   SHD$L_CEFPTR(R6),R6,R0  ; GET ADDR OF FIRST CEB IN TABLE
          51  1C A6   3C  07CC          MOVZWL  SHD$W_CEFMAX(R6),R1     ; GET COUNT OF CEB'S IN TABLE
                 5C   15  07D0          BLEQ    390$                    ; BR IF NO CEF TABLE
          52  15 A5   9A  07D2          MOVZBL  SHB$B_PORT(R5),R2       ; GET PORT # FOR THIS PROCESSOR
          53  08 A0   3C  07D6          MOVZWL  CEB$W_SIZE(R0),R3       ; GET SIZE OF ONE CEB IN BYTES
          54  1D A0   9A  07DA          MOVZBL  CEB$B_PROCCNT(R0),R4    ; GET # OF PROCESSOR PORTS ALLOWED
      54      54 02   78  07DE          ASHL    #2,R4,R4                ; GET # OF BYTES OF SLAVE VA'S
            38 A042   D4  07E2  310$:   CLRL    CEB$L_VASLAVE1(R0)[R2]  ; INDICATE NO SLAVE ENTRY FOR THIS PORT
     55      50  54   C1  07E6          ADDL3   R4,R0,R5                ; COMPUTE ADR OF CEB PLUS SLAVEVA BYTES
```

```
          50    DD  07EA              PUSHL   R0                         ; REMEMBER CEF ADDRESS
          56    DD  07EC              PUSHL   R6                         ; REMEMBER SHD ADDRESS
     56   50    D0  07EE              MOVL    R0,R6                      ; SET CEF ADDRESS
 00000000'GF    16  07F1              JSB     G^EXE$CEBREFLCK            ; ACQUIRE REFCNT LOCK FOR THIS CEF
     38 A542    B4  07F7              CLRW    CEB$L_VASLAVE1(R5)[R2]     ; CLEAR REFERENCE COUNT FOR PORT
     55   6E    D0  07FB              MOVL    (SP),R5                    ; GET ADR OF SHD
  0E 66   01    E0  07FE              BBS     #CEB$V_LOCKED,CEB$L_CEBFL(R6),320$ ; BR IF LOCKED
  0A 66   00    E1  0802              BBC     #CEB$V_VALID,CEB$L_CEBFL(R6),320$  ; BR IF NOT VALID
  1E A6   52    91  0806              CMPB    R2,CEB$B_CREATPORT(R6)     ; IS THIS CEF OWNED BY THIS PORT?
          04    12  080A              BNEQ    320$                       ; BR IF OWNED BY SOME OTHER PORT
     7C A542    B7  080C              DECW    SHD$W_CEFQUOTA(R5)[R2]     ; SUBTRACT ONE FOR THIS CEF OWNERSHIP
                    0810        320$:
       0F 50    E9  0810              BLBC    R0,340$                    ; BR IF UNABLE TO ACQUIRE REFCNT LOCK
  00 66   02    E7  0813              BBCCI   #CEB$V_REFCNTLCK,CEB$L_CEBFL(R6),330$ ; RELEASE REFCNT LOCK
06 0B A6  01    E0  0817        330$: BBS     #CEB$V_PERM,CEB$B_STS(R6),340$ ; BR IF PERMANENT CEF
 00000000'GF    16  081C              JSB     G^EXE$SHMCEBDEL            ; RELEASE CEF IF NOT IN USE BY OTHERS
          56  BED0  0822        340$: POPL    R6                         ; RESTORE SHD ADDRESS
          50  BED0  0825              POPL    R0                         ; RESTORE CEF ADDRESS
     50   53    C0  0828              ADDL2   R3,R0                      ; GET ADR OF NEXT CEB IN TABLE
       B4 51    F5  082B              SOBGTR  R1,310$                    ; LOOP TO INIT NEXT CEB
          3F    BA  082E        390$: POPR    #^M<R0,R1,R2,R3,R4,R5>     ; RESTORE REGISTERS
                    0830
     50   01    D0  0830              MOVL    #1,R0                      ; SET SUCCESS
                05  0833              RSB                                ; RETURN
                    0834
```

```
                         0834          .SBTTL  COMPUTE DATPAGE CRC
                         0834   ;++
                         0834   ;
                         0834   ; DATAPAGE_CRC - COMPUTE DATPAGE CONTSTANT FIELD'S CRC
                         0834   ;
                         0834   ; THIS ROUTINE IS CALLED TO COMPUTE THE CRC OF THE CONSTANT FIELDS
                         0834   ; IN THE SHARED MEMORY DATAPAGE.  THE CRC IS USED TO DETERMINE IF
                         0834   ; THE DATAPAGE IS INTACT.
                         0834   ;
                         0834   ; INPUTS:
                         0834   ;
                         0834   ;       R6 = ADDR OF DATAPAGE
                         0834   ;
                         0834   ; OUTPUTS:
                         0834   ;
                         0834   ;       R0 = CRC OF DATAPAGE CONTSTANT FIELDS
                         0834   ;--
                         0834   DATAPAGE_CRC:                           ; COMPUTE CRC OF DATAPAGE
                         0834          ASSUME  SHD$L_MBXPTR EQ 0        ; ASSUME MBX POINTER IS FIRST
00   F7D2 CF     0B      0834          CRC     AUTODIN,#0,-             ; COMPUTE CRC OF DATAPAGE
       66   38           0839                  #<SHD$L_CRC-SHD$L_MBXPTR>,SHD$L_MBXPTR(R6)
                 05      083B          RSB                              ; RETURN
```

```
                                  083C          .SBTTL  LOAD SHARED MEMORY MAILBOX DRIVER
                                  083C   ;++
                                  083C   ;
                                  083C   ; LOADMBDRIVER - LOAD SHARED MEMORY MAILBOX DRIVER
                                  083C   ;
                                  083C   ; THIS ROUTINE IS CALLED TO LOAD THE MAILBOX DRIVER AND CONNECT A TEMPLATE
                                  083C   ; MAILBOX UCB TO THE SHARED MEMORY.  THE TEMPLATE IS USED TO CREATE THE USER
                                  083C   ; GENERATED MAILBOX UCB'S BY THE $CREMBX SYSTEM SERVICE.
                                  083C   ;
                                  083C   ; INPUTS:
                                  083C   ;
                                  083C   ;       SHR_L_ADP = ADDRESS OF ADAPTER CONTROL BLOCK
                                  083C   ;
                                  083C   ; OUTPUTS:
                                  083C   ;
                                  083C   ;       SHARED MEMORY MAILBOX DRIVER LOADED, DDB, UCB 0, IDB, AND CRB CREATED
                                  083C   ;       AND CONNECTED TO I/O DATABASE.
                                  083C   ;--
                                  083C   LOADMBDRIVER:                                   ; LOAD MAILBOX DRIVER
                            007C  083C          .WORD   ^M<R2,R3,R4,R5,R6>              ;
            5E   D8 AE        9E  083E          MOVAB   -ACF$K_LENGTH(SP),SP            ; ALLOCATE CONFIG CONTROL BLOCK
                 56   5E      D0  0842          MOVL    SP,R6                           ; GET ADDRESS OF BLOCK
         66   003F'CF         D0  0845          MOVL    SHR_L_ADP,ACF$L_ADAPTER(R6)     ; SET ADDRESS OF ADP
                 0A A6        94  084A          CLRB    ACF$B_AUNIT(R6)                 ; SET UNIT #0
                 0B A6        94  084D          CLRB    ACF$B_AFLAG(R6)                 ; SET NO FLAGS
                 0C A6        D4  0850          CLRL    ACF$L_CONTRLREG(R6)             ; SET NO CONTROL REGISTER ADDR
            10 A6   04        B0  0853          MOVW    #PRQ$C_MAILBOX*4,ACF$W_CVECTOR(R6) ; SET MAILBOX VECTOR NUMBER
                 12 A6        B4  0857          CLRW    ACF$W_CUNIT(R6)                 ; SET UNIT #0
            1E A6   01        90  085A          MOVB    #1,ACF$B_CNUMVEC(R6)            ; SET ONE VECTOR
      14 A6   0043'CF         9E  085E          MOVAB   SHR_T_MBDEVNAME,ACF$L_DEVNAME(R6) ; SET ADDRESS OF DEVICE NAME
0046'CF   0008'CF   42 8F     81  0864          ADDB3   #^A7B7,SHR_W_UNIT,SHR_T_MBDEVNAME+3 ; COMPUTE ''CONTROLLER'' NAME
      18 A6   F78F CF         9E  086D          MOVAB   SHR_T_MBDRVNAME,ACF$L_DRVNAME(R6) ; SET ADDRESS OF DRIVER NAME
      1C A6   000C'CF         B0  0873          MOVW    SHR_W_MBXCNT,ACF$W_MAXUNITS(R6) ; SET MAXIMUM NUMBER UNITS
        0000'CF   66          FA  0879          CALLG   (R6),IOGEN$LOADER               ; LOAD THE DRIVER AND CONNECT UCB0
                              04  087E          RET                                     ;
                                  087F
                              087F        .SBTTL  SHOW THE DATA STRUCTURES
                              087F  ;++
                              087F  ;
                              087F  ; SHOW_STRUCT - SHOW THE DATA STRUCTURES
                              087F  ;
                              087F  ; CALLING SEQUENCE:
                              087F  ;
                              087F  ;       $CMEXEC_S SHOW_STRUCT
                              087F  ;
                              087F  ; INPUTS:
                              087F  ;
                              087F  ;       SHR_L_DATAPAGE = ADDRESS OF SHARED MEMORY DATAPAGE
                              087F  ;
                              087F  ; OUPUTS:
                              087F  ;
                              087F  ;       USEFUL INFORMATION ABOUT THE DATA STRUCTURES IS DISPLAYED
                              087F  ;       ON SYS$OUTPUT.
                              087F  ;--
                              087F  SHOW_STRUCT:                                    ; SHOW THE DATA STRUCTURES
                        0FFC  087F        .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; ENTRY MASK
        56   0037'CF    D0    0881        MOVL    SHR_L_DATAPAGE,R6         ; GET ADDR OF DATAPAGE
                              0886        PUT_OUTPUT <DATAPAGE:  !XL>,R6
     57   56    04 A6   C1    08B4        ADDL3   SHD$L_GSDPTR(R6),R6,R7    ; GET ADDR OF GSD TABLE
                              08B9        PUT_OUTPUT <GSD TABLE: !XL>,R7
        57   56    66   C1    08E7        ADDL3   SHD$L_MBXPTR(R6),R6,R7    ; GET ADDR OF MBX TABLE
                              08EB        PUT_OUTPUT <MBX TABLE: !XL>,R7
     57   56    08 A6   C1    0919        ADDL3   SHD$L_CEFPTR(R6),R6,R7    ; GET ADDR OF CEF TABLE
                              091E        PUT_OUTPUT <CEF TABLE: !XL>,R7
     57   56    0C A6   C1    094C        ADDL3   SHD$L_GSBITMAP(R6),R6,R7 ; GET ADDR OF BITMAP
                              0951        PUT_OUTPUT <BITMAP:    !XL>,R7
  57   56   00F0 C6   C1    097F        ADDL3   SHD$Q_PRQ(R6),R6,R7       ; GET ADDR OF PRQ LIST
                              0985        PUT_OUTPUT <PRQ LIST:  !XL>,R7
                              09B3
                          04  09B3        RET                               ; RETURN
                              09B4
                              09B4        .END
```

```
SHARE                          SHARED MEMORY INITIALIZATION      N 2    16-SEP-1984 00:01:41  VAX/VMS Macro V04-00        Page  32
Symbol table                                                             4-SEP-1984 23:05:48  [BOOTS.SRC]SHARE.MAR;1            (1)

$$DESC                          = 000000B5 R       02    FIND_UNIT                     000001DE R       03
$$T2                            = 00000004               GEN$SHARE                     00000148 RG      03
ACB$L_KAST                      = 00000018               GEN$SHR_CEFCNT                000000CC RG      03
ACB$W_SIZE                      = 00000008               GEN$SHR_CEFMAX                000000EE RG      03
ACF$B_AFLAG                     = 0000000B               GEN$SHR_GBLCNT                000000AC RG      03
ACF$B_AUNIT                     = 0000000A               GEN$SHR_GBLMAX                000000DC RG      03
ACF$B_CNUMVEC                   = 0000001E               GEN$SHR_INIT                  00000140 RG      03
ACF$K_LENGTH                    = 00000028               GEN$SHR_MBXCNT                000000BC RG      03
ACF$L_ADAPTER                   = 00000000               GEN$SHR_MBXMAX                000000E5 RG      03
ACF$L_CONTRLREG                 = 0000000C               GEN$SHR_MEMNAME               00000091 RG      03
ACF$L_DEVNAME                   = 00000014               GEN$SHR_POOLC                 000000F7 RG      03
ACF$L_DRVNAME                   = 00000018               GEN$SHR_POOLS                 00000107 RG      03
ACF$W_CUNIT                     = 00000012               GEN$SHR_PRQCNT                00000127 RG      03
ACF$W_CVECTOR                   = 00000010               GEN$SHR_RESET                 0000004A RG      03
ACF$W_MAXUNITS                  = 0000001C               GEN$SHR_START                 00000137 RG      03
ADP$L_CSR                       = 00000000               GEN$SHR_UNIT                  000000A3 RG      03
ADP$L_LINK                      = 00000004               GSD$B_CREATPORT             = 00000052
ADP$L_SHB                       = 00000030               GSD$B_DELETPORT             = 00000053
ADP$W_ADPTYPE                   = 0000000E               GSD$B_LOCK                  = 00000050
ADP$W_TR                        = 0000000C               GSD$B_PROCCNT               = 00000051
AT$_MPM                           ********    X    03    GSD$B_TYPE                  = 0000000A
AUTODIN                           0000000A R       03    GSD$K_SHMGSDLNG             = 00000074
BIT...                          = 00000001               GSD$L_GSDBL                 = 00000004
CEB$B_CREATPORT                 = 0000001E               GSD$L_GSDFL                 = 00000000
CEB$B_DELETPORT                 = 0000001F               GSD$L_PTECNT1               = 00000074
CEB$B_LOCK                      = 0000001C               GSD$V_VALID                 = 00000000
CEB$B_PROCCNT                   = 0000001D               GSD$W_GSTX                  = 00000016
CEB$B_STS                       = 0000000B               GSD$W_SIZE                  = 00000008
CEB$B_TYPE                      = 0000000A               INI$ACONONPAGED               0000020F RG      03
CEB$C_LENGTH                    = 00000038               INI$MPMADP                    ********    X    03
CEB$L_CEBBL                     = 00000004               INIT                          00000217 R       03
CEB$L_CEBFL                     = 00000000               INITLOCK_TIMOUT             = 08F0D180
CEB$L_VASLAVE1                  = 00000038               INITPOLL_TIMOUT             = 02FAF080
CEB$V_LOCKED                    = 00000001               INIT_BITMAP                   000005DD R       03
CEB$V_PERM                      = 00000001               INIT_CEF                      0000068F R       03
CEB$V_REFCNTLCK                 = 00000002               INIT_DATAPAGE                 000003E6 R       03
CEB$V_VALID                     = 00000000               INIT_MAILBOXES                00000676 R       03
CEB$W_SIZE                      = 00000008               INIT_POOL                     00000635 R       03
CHECK_INIT                        00000395 R       03    INIT_PRQ                      00000658 R       03
CONNECT                           0000024E R       03    INIT_STRUCTURES               000005A7 R       03
CONNECTED                         0000025F R       03    IOC$ALLOSPT                   ********    X    03
CONNECT_MEM                       000006D0 R       03    IOC$GL_ADPLIST                ********    X    03
CREATE_SHB                        00000281 R       03    IOGEN$ALLOBLOCK               ********    X    03
DATAPAGE_CRC                      00000834 R       03    IOGEN$LOADER                  ********    X    03
DYN$C_SHB                       = 0000002A               IOGEN$TEST_MEM                ********    X    03
DYN$C_SHMCEB                    = 0000002E               IPL$_HWCLK                  = 00000018
DYN$C_SHMGSD                    = 00000029               LIB$PUT_OUTPUT                ********    X    03
ERR_EXIT                          0000020E R       03    LIB$SIGNAL                    ********    X    03
EXE$CEBREFLCK                     ********    X    03    LOADMBDRIVER                  0000083C R       03
EXE$GL_CONFREGL                   ********    X    03    LOCK_DATAPAGE                 0000035C R       03
EXE$GL_LOCKRTRY                   ********    X    03    MA$INITIAL                    ********    X    03
EXE$GL_NUMNEXUS                   ********    X    03    MA$REQUEST                    ********    X    03
EXE$GL_RPB                        ********    X    03    MAP_DATAPAGE                  000002CC R       03
EXE$GL_SHBLIST                    ********    X    03    MAP_STRUCTURES                00000562 R       03
EXE$GQ_SYSTIME                    ********    X    03    MBX$B_CREATPORT             = 00000009
EXE$SHMCEBDEL                     ********    X    03    MBX$B_FLAGS                 = 00000008
EXIT                              0000027D R       03    MBX$K_LENGTH                = 00000030
```

```
                                                          B  3
SHARE                            SHARED MEMORY INITIALIZATION          16-SEP-1984 00:01:41  VAX/VMS Macro V04-00        Page  33
Symbol table                                                            4-SEP-1984 23:05:48  [BOOTS.SRC]SHARE.MAR;1            (1)

MBX$W_REF                        = 0000000C                 SHD$M_INITLCK                = 00000001
MBX$W_UNIT                       = 0000000A                 SHD$Q_INITTIME               = 00000030
MPM$C_PORTS                      = 00000004                 SHD$Q_POOL                   = 000000F8
MPM$L_CSR                        = 00000000                 SHD$Q_PRQ                    = 000000F0
MPM$L_INV                        = 0000000C                 SHD$Q_PRQWRK                 = 00000100
MPM$L_MR                         = 0000001C                 SHD$T_NAME                   = 00000020
MPM$S_CSR_PORT                   = 00000002                 SHD$V_INITLCK                = 00000000
MPM$S_INV_MEMSZ                  = 00000003                 SHD$V_MBXLCK                 = 00000003
MPM$S_INV_STADR                  = 0000000B                 SHD$W_CEFMAX                 = 0000001C
MPM$S_MR_UNIT                    = 00000002                 SHD$W_CEFQUOTA               = 0000007C
MPM$V_CSR_PORT                   = 00000000                 SHD$W_GSDMAX                 = 00000018
MPM$V_INV_MEMSZ                  = 00000010                 SHD$W_GSDQUOTA               = 0000003C
MPM$V_INV_STADR                  = 00000014                 SHD$W_MBXMAX                 = 0000001A
MPM$V_MR_UNIT                    = 0000000E                 SHD$W_MBXQUOTA               = 0000005C
NDT$_MPM0                        = 00000040                 SHD$W_POLL                   = 000000A6
NDT$_MPM3                        = 00000043                 SHD$W_PRQWAIT                = 000000A4
NO_INIT                            000003E3 R      03       SHD$W_RESAVAIL               = 000000C8
POLL                               0000039A R      03       SHD$W_RESSUM                 = 000000E8
PR$_IPL                          = 00000012                 SHD$W_RESWAIT                = 000000A8
PR$_TBIS                         = 0000003A                 SHOW_STRUCT                    0000087F R      03
PRQ$C_MAILBOX                    = 00000001                 SHR_B_OPTIONS                  00000026 R      02
PRQ$C_MINLENGTH                  = 00000040                 SHR_L_ADP                      0000003F R      02
PTE$C_ERKW                       = 30000000                 SHR_L_CEFSIZE                  00000033 R      02
PTE$M_VALID                      = 80000000                 SHR_L_DATAPAGE                 00000037 R      02
PTE$S_PFN                        = 00000015                 SHR_L_GSDSIZE                  0000002F R      02
PTE$V_PFN                        = 00000000                 SHR_L_MEMPFN                   0000002B R      02
RPB$L_BOOTR5                     = 00000030                 SHR_L_MEMSIZE                  00000027 R      02
RPB$L_MEMDSC                     = 000000BC                 SHR_L_POOLBCNT                 00000016 R      02
RPB$M_MPM                        = 00000800                 SHR_L_POOLBSIZ                 0000001A R      02
RPB$M_USEMPM                     = 00001000                 SHR_L_PRQCNT                   0000001E R      02
RSN$_MAX                         = 0000000F                 SHR_L_SHDPTE                   0000003B R      02
SHARE                              00000185 R      03       SHR_L_START                    00000022 R      02
SHB$B_FLAGS                      = 0000000B                 SHR_OPT_M_INIT               = 00000001
SHB$B_NEXUS                      = 00000014                 SHR_OPT_V_INIT               = 00000000
SHB$B_PORT                       = 00000015                 SHR_Q_MEMNAME                  00000000 R      02
SHB$B_TYPE                       = 0000000A                 SHR_T_MBDEVNAME                00000043 R      02
SHB$K_LENGTH                     = 00000020                 SHR_T_MBDRVNAME                00000000 R      03
SHB$L_ADP                        = 0000001C                 SHR_VALUES                     00000000 R      02
SHB$L_BASGSPFN                   = 00000010                 SHR_W_CEFCNT                   0000000E R      02
SHB$L_DATAPAGE                   = 00000004                 SHR_W_CEFQUO                   00000014 R      02
SHB$L_LINK                       = 00000000                 SHR_W_GBLCNT                   0000000A R      02
SHB$L_POOLEND                    = 00000018                 SHR_W_GBLQUO                   00000010 R      02
SHB$L_REFCNT                     = 0000000C                 SHR_W_MBXCNT                   0000000C R      02
SHB$M_CONNECT                    = 00000001                 SHR_W_MBXQUO                   00000012 R      02
SHB$V_CONNECT                    = 00000000                 SHR_W_UNIT                     00000008 R      02
SHB$W_SIZE                       = 00000008                 SIZ...                       = 00000001
SHD$B_FLAGS                      = 0000009F                 SS$_NORMAL                   = 00000001
SHD$B_INITLCK                    = 0000009D                 STS$K_ERROR                  = 00000002
SHD$B_PORTS                      = 0000009C                 STS$K_SUCCESS                = 00000001
SHD$K_LENGTH                     = 00000180                 STS$S_SEVERITY               = 00000003
SHD$L_CEFPTR                     = 00000008                 STS$V_SEVERITY               = 00000000
SHD$L_CRC                        = 00000038                 SYS$CMEXEC                     ********  GX    03
SHD$L_GSBITMAP                   = 0000000C                 SYS$CMKRNL                     ********  GX    03
SHD$L_GSDPTR                     = 00000004                 SYS$FAO                        ********   X    03
SHD$L_GSPAGCNT                   = 00000010                 SYSG$_BADCHKSUM              = 007C8052
SHD$L_GSPFN                      = 00000014                 SYSG$_BADPARAM               = 007C804A
SHD$L_MBXPTR                     = 00000000                 SYSG$_INCMEMNAM              = 007C805A
```

```
SYSG$_NOSUCHMEM                 = 007C8042
SYSG$_SHMDBLUSE                 = 007C810A
SYSG$_SPTFULL                   = 007C8022
TPA$L_NUMBER                    = 0000001C
TPA$L_TOKENCNT                  = 00000010
UNLOCK_DATAPAGE                   0000038E R      03
UNLOCK_EXIT                       0000027A R      03
VA$M_SYSTEM                     = 80000000
VA$S_VPN                        = 00000015
VA$V_VPN                        = 00000009

                                              +----------------+
                                              ! Psect synopsis !
                                              +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           00000000  (    0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
NONPAGED_DATA                   000000CB  (  203.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD     WRT NOVEC QUAD
NONPAGED_CODE                   000009B4  ( 2484.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC LONG

                                           +------------------------+
                                           ! Performance indicators !
                                           +------------------------+

Phase                       Page faults    CPU Time        Elapsed Time
-----                       -----------    --------        ------------
Initialization                       35    00:00:00.07     00:00:00.36
Command processing                  134    00:00:00.68     00:00:04.53
Pass 1                              456    00:00:18.00     00:00:33.60
Symbol table sort                     0    00:00:02.18     00:00:04.24
Pass 2                              259    00:00:04.50     00:00:09.51
Symbol table output                  28    00:00:00.23     00:00:00.62
Psect synopsis output                 1    00:00:00.03     00:00:00.03
Cross-reference output                0    00:00:00.00     00:00:00.00
Assembler run totals                915    00:00:25.70     00:00:52.90

The working set limit was 1800 pages.
102872 bytes (201 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1465 non-local and 71 local symbols.
1355 source lines were read in Pass 1, producing 24 object records in Pass 2.
44 pages of virtual memory were used to define 41 macros.

                                          +--------------------------+
                                          ! Macro library statistics !
                                          +--------------------------+

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[BOOTS.OBJ]BOOTS.MLB;1                     0
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        24
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     13
TOTALS (all libraries)                                37

1619 GETS were required to define 37 macros.
```

```
There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SHARE/OBJ=OBJ$:SHARE MSRC$:SHARE/UPDATE=(ENH$:SHARE)+EXECML$/LIB+LIB$:BOOTS.MLB/LIB
```

0039 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

READLBN
LIS

RTFILREAD
LIS

SHARE
LIS

QUSS
LIS

RXBTDRIVR
LIS

MBBTDRIVR
LIS

SCSLOADER
LIS

RMSCONIO
LIS

READDRIV
LIS

PABTDRIVR
LIS

PUBTDRIVR
LIS

PUTERROR
LIS

READPRMPT
LIS

0040 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY